VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV RADIOELEKTRONIKY
DEPARTMENT OF RADIO ELECTRONICS

AUTENTIZACE RF VYSÍLAČŮ NA ZÁKLADĚ NEDOKONALOSTÍ RÁDIOVÉHO ŘETĚZCE
RF TRANSMITTER AUTHENTICATION BASED ON FRONT-END IMPAIRMENTS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Kristina Youssefová**
AUTHOR

VEDOUCÍ PRÁCE — **prof. Ing. Roman Maršálek, Ph.D.**
SUPERVISOR

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Elektronika a komunikační technologie**

Ústav radioelektroniky

Studentka: Bc. Kristina Youssefová ***ID:*** 205961
Ročník: 2 ***Akademický rok:*** 2020/21

NÁZEV TÉMATU:

Autentizace RF vysílačů na základě nedokonalostí rádiového řetězce

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s parametry reálných rádiových front-endů jako např. nelinearita zesilovače, nesymetrie IQ modulátoru, stejnosměrný ofset. Prostudujte používané metody pro autentizaci rádiového vysílače na základě odhadu zkreslení vysílaného signálu rádiovým front-endem. Zaměřte se na základní metody strojového učení jako například Support Vector Machines (SVM) nebo dnes hojně používané neuronové sítě. V prostředí MATLAB implementujte vybranou metodu pro autentizaci a ověřte ji na sadě dat z měření deseti softwarově definovaných rádií USRP dostupných na UREL.

Na základě metody Support Vector Machines (SVM) vytvořte demonstrační aplikaci v prostředí MATLAB, která umožní ověřit identitu vysílače, jehož signál je přijímán, a vyhodnotit zda patří do skupiny známých vysílačů. Pro ověření využijte měřených záznamů sady softwarově definovaných rádií USRP. Metodu SVM srovnejte s vybranou metodou klasifikace využívající neuronovou síť. Využijte dostupné toolboxy v prostředí MATLAB.

DOPORUČENÁ LITERATURA:

[1] WANG, Xueli, Yufeng ZHANG, Hongxin ZHANG, Xiaofeng WEI a Guangyuan WANG. Identification and authentication for wireless transmission security based on RF-DNA fingerprint. EURASIP Journal on Wireless Communications and Networking [online]. 2019, 2019(1) [cit. 2020-05-18]. DOI: 10.1186/s13638-019-1544-8. ISSN 1687-1499.

[2] WU, Qingyang, Carlos FERES, Daniel KUZMENKO, Ding ZHI, Zhou YU, Xin LIU a Xiaoguang 'LEO' LIU. Deep learning based RF fingerprinting for device identification and wireless security. Electronics Letters. 2018, 54(24), 1405-1407

Termín zadání: 8.2.2021 ***Termín odevzdání:*** 20.5.2021

Vedoucí práce: prof. Ing. Roman Maršálek, Ph.D.

prof. Ing. Tomáš Kratochvíl, Ph.D.
předseda rady studijního programu

Abstrakt

Tato práce se zaměřuje na klasifikaci vysokofrekvenčních vysílačů v závislosti na nedokonalostech jejich komponent pomocí algoritmu strojového učení. Práce je rozdělena na dvě části - teoretickou a praktickou.V teoretické části je nejprve popsána základní struktura vysílače s přímou konverzí a jsou uvedeny nedokonalosti rádiového front-endu, které mohou být využity ke klasifikaci. Dále jsou vysvětleny vybrané metody strojového učení s učitelem, zejména metoda support vector machines a neuronové sítě. Praktická část se zabývá implementací a dosaženými výsledky těchto dvou metod v prostředí MATLAB na problému klasifikace rádiových front-endů.

Klíčová slova

RF front-end, I/Q nevyvážení, CFO, konstelace původ ofset, strojové učení, Klasifikace, SVM, OAO, neuronová síť, neuron, Matlab

Abstract

This work focuses on classifying Radio Frequency transmitters depending on their Radiofrequency imperfections by using a machine learning algorithm. The thesis is divided into two parts – theoretical and practical. The theoretical part can be divided into three branches. In the first branch, an overview of the RF transmitter with direct conversion is given. In the second one, Possible imperfections in the radio front-end are studied. In the third one, some supervised machine learning algorithms were explained. A detailed explanation of the support vector machine and neural network algorithms is given. The practical part deals with the implementation of support vector machines and neural networks in the MATLAB program and the evaluation of results.

Keywords

RF front-end, I/Q imbalance, CFO, constellation origin offset, machine learning, classification, SVM, OAO, neural network, neuron, Matlab

Rozšířený abstrakt

S nárůstem počtu bezdrátových zařízení se otázka ověření identity komunikujících zařízení stává stále důležitější. Tato práce se zaměřuje na klasifikaci vysokofrekvenčních vysílačů v závislosti na nedokonalostech jejich komponent pomocí algoritmů strojového učení. Ačkoli by jejich nominální parametry měly být shodné, mohou se jednotlivé vysokofrekvenční vysílače svými vlastnostmi lišit. Významnou odlišností mohou být vzhledem k tolerancím součástek, skutečné (nedokonalé) parametry rádiového front-endu. Tyto nedokonalosti (například nelinearita výkonového zesilovače, I/Q nevyvážení, posun počátku konstelačního diagramu, posun nosné frekvence) lze použít ke klasifikaci dat konkrétního vysílače mezi ostatními zástupci.

Klasifikace dat je běžným úkolem metod tzv. strojového učení. V první fázi probíhá učení/trénování za pomoci trénovací sady dat. Následně je výkonost metody testována na testovací sadě dat, kdy v našem případě probíhá klasifikace neznámých vysílačů do jednotlivých tříd.

V teoretické části práce je nejprve popsána základní struktura vysílače s přímou konverzí a jsou uvedeny nedokonalosti rádiového front-endu, které mohou být využity ke klasifikaci jednotlivých vysílačů. Algoritmy strojového učení se dělí na algoritmy strojového učení s učitelem a algoritmy strojového učení bez učitele. Algoritmy strojového učení s učitelem, používají trénování na označených datech, zatímco algoritmy strojového učení bez učitele předpokládají trénování na datech neoznačených. V této diplomové práci jsou vysvětleny vybrané metody strojového učení s učitelem, zejména metoda tzv. support vector machines (SVM) a metoda založená na umělých neuronových sítích.

Praktická část práce se zabývá implementací těchto dvou metod v prostředí MATLAB a jejich aplikací na problému klasifikace rádiových front-endů, a uvádí dosažené výsledky na sadě osmi front-endů softwarově definovaného rádia USRP.

Postupně byly vytvořeny tři scénáře pro algoritmus SVM: binární klasifikace využívající umělá data, binární klasifikace využívající skutečný soubor dat dvou samostatných vysílačů, klasifikace více tříd pomoci SVM využívající skutečný soubor dat tří vysílačů a poté klasifikace více tříd využívající skutečná datová sady osmi vysílačů. Různé kombinace vlastností vysílačů (tj. např. výše uvedené amplitudové a fázové nevyvážení kvadraturního modulátoru nebo kmitočtový ofset) pak tvoří dvourozměrný, resp. vícerozměrný prostor příznaků SVM klasifikátoru. Tímto způsobem je zjištěno, jaký z hardwarových parametrů, resp. jaká jejich kombinace, umožňuje nejspolehlivější klasifikaci mezi všemi vysílači.

Algoritmus neuronové sítě použitý v naší práci představuje tzv. mělkou (shallow) neuronovou síť. Nejprve bylo zjištěno jak závisí úspěšnost klasifikace na počtu neuronů, a byl určen optimální počet neuronů skryté vrstvy. Podobně jako v případu SVM, byly zkoumány různé kombinace příznaků vysílače ve dvourozměrném a trojrozměrném prostoru. Tímto způsobem bylo zjištěno, které příznaky umožnují nejspolehlivější klasifikaci pomocí neuronové sítě. V závěru byly obě metody klasifikace – SVM a neuronové sítě, vzájemně porovnány a byly diskutovány dosažené výsledky.

Bibliografická citace:

YOUSSEFOVÁ, *Kristina*. *Autentizace RF vysílačů na základě nedokonalostí rádiového řetězce* [online]. Brno, 2021. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav radioelektroniky. Vedoucí práce *prof. Ing. Romanu Maršálkovi,Ph.D.*

Jméno a příjmení studenta:	*Kristina Youssefová*
VUT ID studenta:	*205961*
Typ práce:	*Diplomová práce*
Akademický rok:	*2020/21*
Téma závěrečné práce:	*Autentizace RF vysílačů na základě nedokonalostí rádiového řetězce*

Prohlášení

Prohlašuji, že svou diplomovou práci na téma Autentizace RF vysílačů na základě nedokonalostí rádiového řetězce jsem vypracovala samostatně pod vedením vedoucí/ho diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Tretního zákoníku č. 40/2009 Sb.

V Brně dne: **20. Květen 2020**

……
podpis autora

Poděkování

Děkuji vedoucímu diplomové práce prof. Ing. Romanu Maršálkovi,Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne: **20. Květen 2020**

……
podpis autora

Content

1. Introduction 13
2. Radio Frequency transmitter 14
2.1 Radiofrequency front-end 14
2.1.1 Local oscillator 14
2.1.2 Quadrature Mixer (quadrature upconverter) 15
2.1.3 Power amplifier 15
2.2 Radio Frequency Front-End Imperfections 15
2.2.1 I/Q imbalance 15
2.2.1.1 I/Q Gain Imbalance (GI) 15
2.2.1.2 I/Q phase imbalance (PI) 15
2.2.2 Power Amplifier Nonlinearity 16
2.2.3 Carrier frequency offset 16
2.2.4 Constellation Origin offset 17
3. Classification Methods 18
3.1 Support Vector Machines 18
3.1.1 Binary classification 18
3.1.1.1 SVM in linear separable cases 18
3.1.1.2 SVM in nonlinear separable case 20
3.1.1.3 Non-linear classification 20
3.1.2 Multi classification SVM 21
3.2 K-Nearest Neighbor (KNN) classification 22
3.3 Neural Networks 23
4. MEASUREMENT Setup 26
5. Implementing SVM and Neural Network with Matlab 28
5.1 The implementation of the SVM algorithm in Matlab 28
5.1.1 scenario 1: Build a simple SVM with artificial data 28
5.1.2 scenario 2: Build a SVM code with real data for binary classification 28
5.1.3 scenario 3: Build a SVM code with real data for multi-class classification
……30

5.2 The implementation of Neural network algorithm in Matlab 32
5.3 Results 32
6. CONCLUSION 49

List of Figures

Figure.2-1: Simplified block diagram of RF transmitter with direct conversion 14
Figure.2-2: Constellation mapping for QPSK with gain imbalance 16
Figure.2-3: Constellation mapping for QPSK with phase imbalance 16
Figure.2-4: Constellation mapping for QPSK with Origin offset 17
Figure.3-1: Linear SVM binary classification: Hard margin 19
Figure.3-2: Linear SVM binary classification: Soft margin 20
Figure.3-3: Binary SVM classification with non-linear decision boundary: RBF kernel [10] 21
Figure.3-4:Non-linear SVM classification using kernel trick [11] 21
Figure.3-5: one-vs-all (based on [13]) 22
Figure.3-6: one-vs-one (based on [13]) 22
Figure.3-7: KNN classification 23
Figure.3-8: The neuron structure [15] 23
Figure.3-9: The activation functions: (a) Sigmoid function. (b) Tanh function. (c) ReLU function 24
Figure.3-10: Simplified diagram of Neural network : (a) shallow neural network. (b) Deep neural network 25
Figure.3-11: Simplified diagram of NN training algorithm 25
Figure.4-1: Laboratory workplace for sending and receiving the Rf signal [18] 26
Figure.5-1: SVM binary classification with artificial data(linear separable case)... 33
Figure.5-2: SVM binary classification with artificial data (linear non-separable case) 33
Figure.5-3: ROC curve for better and worse SVM binary classifier with artificial data 34
Figure.5-4: Binary classification for measured data of RF1 and RF4 (linear separable case) 34
Figure.5-5: Binary classification for measured data of RF1 and RF4. The CFO and the gain imbalance have been used as the SVM features 35
Figure.5-6: Binary classification for measured data of RF1 and RF4. The CFO, the gain imbalance, and the quadrature error have been used as the SVM features..... 35

Figure.5-7: ROC curves for binary SVM classification: (a) between RF1- RF4. (b) between RF3- RF4. (c) between RF1- RF2 36
Figure.5-8: Confusion matrix for binary SVM classification: (a) between RF1- RF4. (b) between RF3- RF4. (c) between RF1- RF2 37
Figure.5-9: Multi-class classification for real data of RF1, RF4, and RF7. 38
Figure.5-10: The data distribution for the eight transmitters and their confusion matrix. The CFO and the gain imbalance have been used as the SVM features 39
Figure.5-11: The data distribution for the eight transmitters and their confusion matrix. The CFO and the quadrature error have been used as the SVM features 40
Figure.5-12: The data distribution for the eight transmitters and their confusion matrix. The CFO and the origin offset have been used as the SVM features..... 40
Figure.5-13: The data distribution for the eight transmitters and their confusion matrix. The gain imbalance, CFO, and the quadrature error have been used as the SVM features 41
Figure.5-14: The data distribution for the eight transmitters and their confusion matrix. The origin offset, CFO, and the quadrature error have been used as the SVM features..... 41
Figure.5-15: The data distribution for the eight transmitters and their confusion matrix. The gain imbalance and CFO have been used as the SVM features 42
Figure.5-16: The data distribution for the eight transmitters and their confusion matrix. The gain imbalance, CFO, and quadrature error have been used as the SVM features..... 43
Figure.5-17: Simplified diagram of MaxWins strategy. The red block selects such a transmitter having the maximum number of occurrences accumulated overall binary learners NLall 43
Figure.5-18: Neural network performance as a function of the number of neurons in the hidden layer..... 44
Figure.5-19: The data distribution for the eight transmitters and their confusion matrix. The CFO and the gain imbalance have been used as the neural network features..... 45

Figure.5-20: The data distribution for the eight transmitters and their confusion matrix. The CFO and the quadrature error have been used as the neural network features 45
Figure.5-21: The data distribution for the eight transmitters and their confusion matrix. The CFO and the origin offset have been used as the Neural network features 46
Figure.5-22: The data distribution for the eight transmitters and their confusion matrix. The CFO, gain imbalance and quadrature error have been used as the Neural network features 47
Figure.5-23: The data distribution for the eight transmitters and their confusion matrix. The CFO, origin offset, and quadrature error have been used as the Neural network features 47
Figure.5-24: The data distribution (0.5 % of the total dataset) for the eight transmitters and their confusion matrix. The CFO, origin offset, and quadrature error have been used as the Neural network features 48

List of Tables

Tab.4-1: The combination of the mainboard and front-end board..... 26
Tab.4-2: The main parameters that were set for the transmitted signal and the VSA
..... 27
Tab.5-1: The true positive rate and the false positive rate..... 30
Tab.5-2: The classes that each binary classifier works with..... 39

1. INTRODUCTION

The wireless devices classification based on the transmitter imperfections is one of the promising ways to provide additional security to future wireless communication networks [1]. These imperfections (such as power amplifier nonlinearity, IQ modulator imbalance, constellation origin offset, carrier frequency offset) can be used to classify the data of a specific transmitter amongst the others representatives [1]. The main source of these imperfections are the analog devices (digital to analog converter, mixer, local oscillator, and power amplifier) that are present in the radio frequency front-end.

The main goal of this thesis is to classify radiofrequency transmitters depending on their radiofrequency imperfections using a selected machine learning algorithm. Various supervised machine learning techniques in the radiofrequency domain can be used for this purpose. In this thesis, the support vector machine and the neural network algorithm are used.

This thesis is divided into six chapters. The primary radiofrequency transmitter chain is briefly described in section 2. This section also studies the possible imperfections of a typical direct conversion front-end board and describes these imperfections. Section 3 deals with the presentation of the classification methods and describes some supervised learning algorithms, for instance, Support vector machines, k-nearest-neighbor, and neural network algorithms. In this section, the main description is based on the SVM and the neural network classifier, that will be used to classify the data of transmitters. A laboratory measurement workplace used to measure the RF imperfections and to classify them is briefly presented in section 4. As the classification was implemented in the Matlab environment, section 5 thus includes a brief description of the most important functions of this software. Results from Matlab and their discussion are also presented in section 5. Finally, this thesis is concluded in section 6.

2. RADIO FREQUENCY TRANSMITTER

The radio frequency transmitter creates the radio frequency signal and sends it to the antenna (see an example of direct conversion transmitter architecture in Figure.2-1). It consists of a digital modulator that produces the in-phase (I) and quadrature (Q) components. The I and Q components subsequently go through the digital to analog converter (D/A) that converts a digital signal into an analog signal. The low pass filter (LPF) only allows signals below its corner frequency to pass. The main application of this filter is to suppress the extra high-frequency images that are created by the D/A converter [2]. The components on the right colored pink present the Radio Frequency (RF) front end. The output of this transmitter is presented by this equation [2]:

$$S(t) = I(t).\cos(2\Pi f_c t) - Q(t).\sin(2\Pi f_c t) \qquad \textbf{(2.1)}$$

where f_c is the local oscillator (LO) frequency.

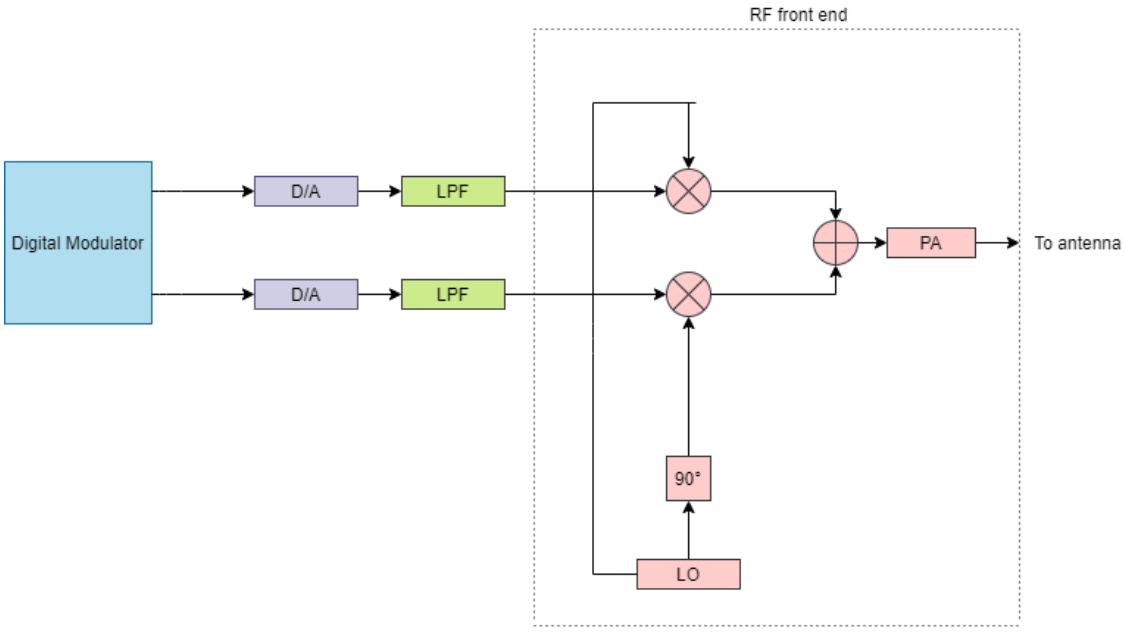


Figure.2-1: Simplified block diagram of RF transmitter with direct conversion

2.1 Radiofrequency front-end

RF front-end is a module that includes all the components between the D/A converter and the antenna. It consists of a Local Oscillator, mixer, and Power Amplifier.

2.1.1 Local oscillator

The Local Oscillator (LO) generates a cosine wave and a copy of that signal that is phase-shifted (in the ideal case) by 90°.

2.1.2 Quadrature Mixer (quadrature upconverter)

The mixer mixes the signal from the local oscillator with the incoming I signal and the 90° shifted signal from the local oscillator with the Q signal to convert them into RF signals.

2.1.3 Power amplifier

The power amplifier (PA) is a device, which is responsible for amplifying the transmitted signal to a necessary power level for transmission to the receiver [2].

2.2 Radio Frequency Front-End Imperfections

The RF signal is influenced by the hardware imperfections of the transmitter. These imperfections exist because of the nonlinear behavior of the RF frontend of the transmitter and this results in limiting the quality of the signal and degrading it [2].

Good examples of such imperfections are:

2.2.1 I/Q imbalance

This problem appears in analog quadrature mixers that are present in the RF transmitter. This is because they are very sensitive to the problem of imbalance between phase and quadrature branches [2].

I/Q imbalance can be divided into two imbalances:

2.2.1.1 I/Q Gain Imbalance (GI)

It is caused by the gain difference between I and Q branches (see Figure.2-2). The gain imbalance is given by [2]:

$$20.\log_{10}\frac{I}{Q} \qquad \textbf{(2.1)}$$

2.2.1.2 I/Q phase imbalance (PI)

The I/Q phase imbalance occurs when the phase shift between I and Q branches differs from the theoretical value of 90 (see Figure.2-3).

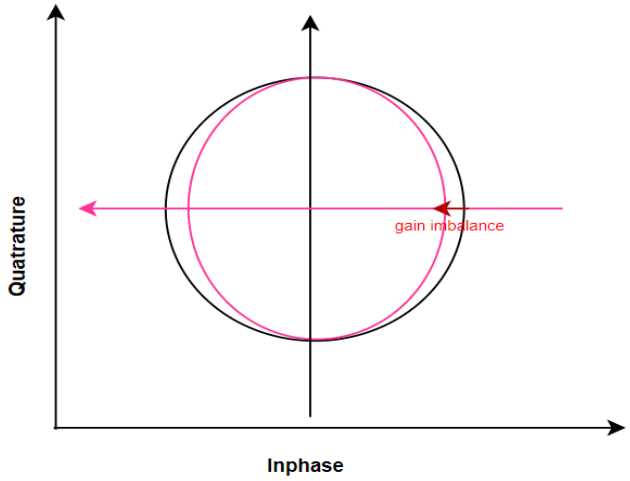


Figure.2-2: Constellation mapping for QPSK with gain imbalance

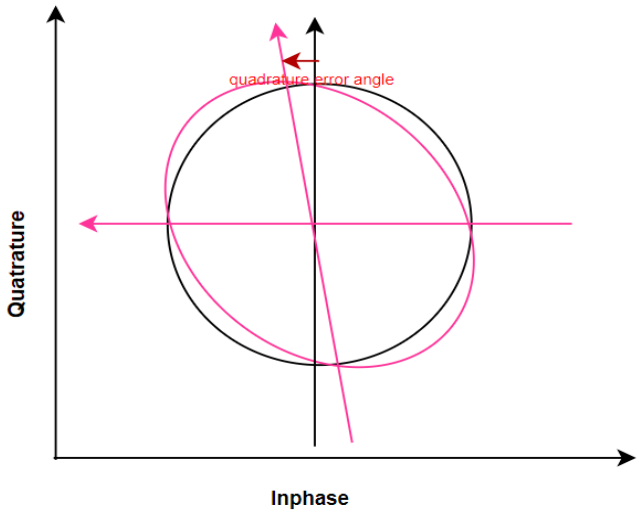


Figure.2-3: Constellation mapping for QPSK with phase imbalance

2.2.2 Power Amplifier Nonlinearity

As we mentioned before, the amplifier amplifies the signal to overcome the path loss, and thus the signal can reach the receiver with enough power for successful detection. The maximum efficiency for modern PA is achieved closer to its saturation point [3]. This results in unwanted effects on the signal that can degrade it.

2.2.3 Carrier frequency offset

The Carrier Frequency offset (CFO) exists because of the frequency mismatch between transmitter and receiver local oscillators [4]. As a result, the received signal will be slightly shifted in frequency. This affects the work of some systems. For instance, in the OFDM system, the orthogonality among subcarriers is not maintained [4].

2.2.4 Constellation Origin offset

The signal from the local oscillator leaks to the input of the power amplifier, in which it is amplified. At the receiver side, this results in shifting the center of the constellation diagram of the origin, on Inphase and Quadrature axes.

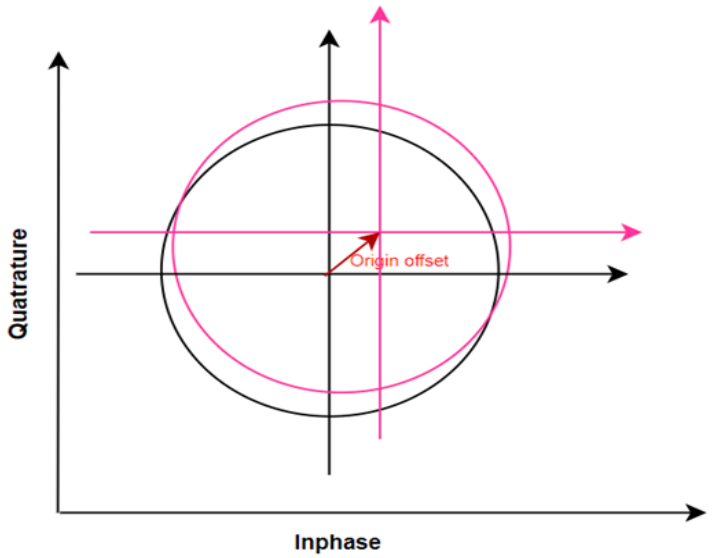


Figure.2-4: Constellation mapping for QPSK with Origin offset

3. CLASSIFICATION METHODS

With the increase in the number of connected wireless devices, transmitter identification has become an important tool to stop malicious transmitters from impersonation [5]. Each radio frequency transmitter differs in its characteristics. They have different RF frontend imperfections that can be used for classifying. Classifying data is a common task in machine learning. At first, they would learn the data and then classify them into their radio transmitter. To do so various machine learning algorithms can be applied. Machine learning algorithms are divided into supervised learning algorithms and unsupervised learning algorithms [6]. Supervised algorithms are when you train the machine learning module with labeled data while unsupervised algorithms are when you train the module with unlabeled data [6]. In this thesis, we will talk about supervised learning algorithms. For instance, Support vector machines, k-nearest-neighbor, and neural networks algorithms. The primary focus will be on support vector machines and neural network algorithms.

3.1 Support Vector Machines

Support vector machine (SVM) is a supervised machine learning algorithm used for classification [7]. The algorithm searches for the optimal hyperplane that best separates the data into the classes and that means all data points of one class will be separated from those of the other classes. According to the number of classes, SVM classification can be divided into Binary classification and Multi-class classification.

3.1.1 Binary classification

In this case, the data has exactly two classes and the hyperplane separating the data can be either linear (linear classification) or nonlinear (non-linear classification).

3.1.1.1 SVM in linear separable cases

Let us assume we have two types of data presented in two-dimensional space. The pink data points represent the first class with label 1 and the blue data points represent the second class with label -1. The optimal hyperplane (a line for a two-dimensional space) can be achieved by choosing the line that has the maximum distance from the nearest data points of both classes. The nearest data points are called support vectors. The distance between the hyperplane and the support vectors is called the Margin (see Figure.3-1). From the figure, it is evident that the data is linearly separable and that means there are no data points in the wrong class. This case is known as a hard margin [7].

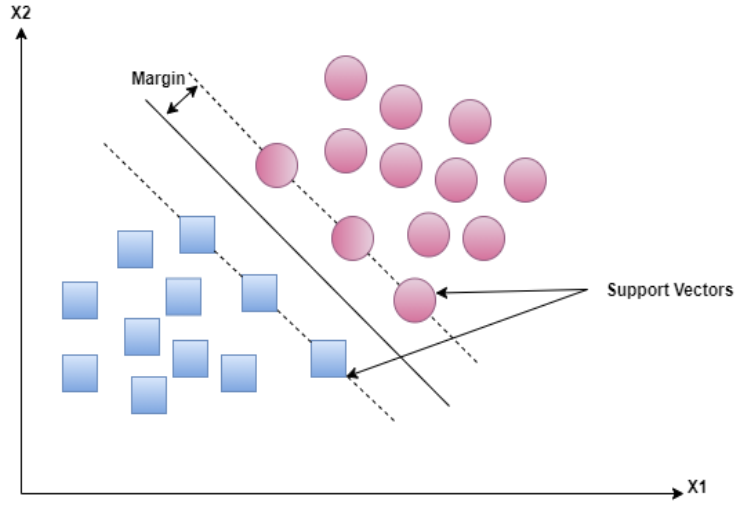


Figure.3-1: Linear SVM binary classification: Hard margin

From the figure, the equation of the line can be written as follows [8]:

$$(w1.x1) + (w2.x2) + b = 0 \Rightarrow w.x + b = 0 \qquad (\mathbf{3.1})$$

where w= (w1,w2) is the normal vector to the line, x= (x1,x2) is the total data points considering x1 and x2 as features. b is the offset of the line from the origin. The distance between the line passing through support vectors of the first class and the line passing through support vectors of the second class (marked as dotted lines) is known as the classification interval [8]. The dotted lines are given by [8]:

$$w.x + b = 1 \qquad \mathbf{(3.2)}$$

$$w.x + b = -1 \qquad \mathbf{(3.3)}$$

The classification interval can be deduced according to [8]:

$$(x_p - x_b).\frac{w}{\| w \|} = \frac{x_p.w - x_b.w}{\| w \|}$$

$$= \frac{(1-b)-(-1-b)}{\|w\|} = \frac{2}{\|w\|} \qquad \mathbf{(3.4)}$$

where $\frac{w}{\|w\|}$ is the unit vector of w, x_p is the support vector of the pink cluster, and x_b is the support-vector of the blue cluster.

To find the best hyperplane with the maximum margin, the classification interval should be maximized. In addition, some constraints should be met to classify the data points correctly [8]:

$$w.x + b \geq 1 \qquad \mathbf{(3.5)}$$

$$w.x + b \leq -1 \qquad (\mathbf{3.6})$$

These equations mean that all data points on or above the hyperplane would be classified to the first class with label 1 and all data points on or below the hyperplane

would be classified to the second class with label -1 [8]. To generalize these two equations [8]:

$$y_i \,.\, (w.x_i + b) \geq 1 \qquad \textbf{(3.7)}$$

where i=1,2,3,……n and y_i is the label's value.

Hence to find the optimal hyperplane, two conditions must be met [8]:

- $\|w\|$ or $\frac{1}{2} \,.\, \| w \|^2$ must be minimized
- $y_i.(w.x_i + b) \geq 1$ must be met.

3.1.1.2 SVM in nonlinear separable case

In the linearly separable case, all data points are classified correctly. But in reality, data points are never completely separable. So, some data points will not be classified correctly, but the margin is kept as wide as possible so that other data points can still be classified correctly [7]. This case is known as the soft margin [7]. So, if there were data points from the pink cluster within the blue data points cluster, then the dataset becomes nonlinear separable (see Figure.3-2). In this case, to find the optimal line that separates the classes, these conditions must be met [8]:

$$y_i.(w.x_i + b) \geq 1 - \xi_i \qquad \textbf{(3.8)}$$

$$min\,(\frac{1}{2}.\| w \|^2 + C.\sum \xi_i\,) \qquad \textbf{(3.9)}$$

where *C* is the penalty coefficient and ζ_i is the so-called slack variable and it represents point classification error.

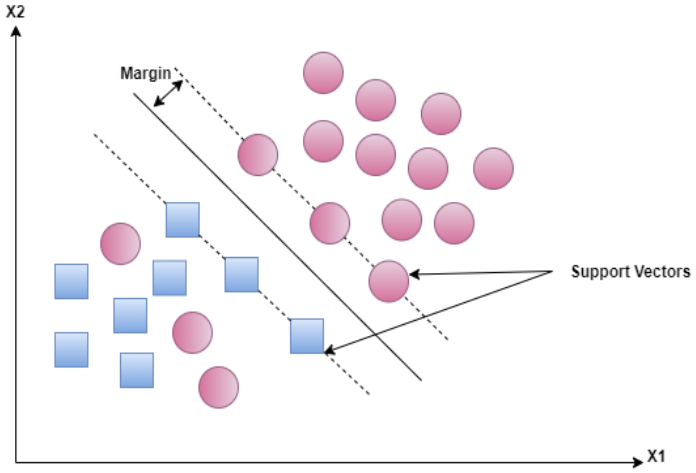


Figure.3-2: Linear SVM binary classification: Soft margin

3.1.1.3 Non-linear classification

As mentioned above, some data can be separated linearly even though they include noisy data points. But In many cases, the dataset can not be separated linearly. This problem can be solved by using a non-linear boundary decision. This curved decision boundary is

more flexible to separate the classes without any errors. To do so non-linear kernel functions are used [9]. These kernel functions apply some type of transformations to the features and then create new features, thus the non-linear boundary can be found [9]. The most popular kernel functions are the Polynomial and Radial Basis Function(RBF). (see Figure.3-3).

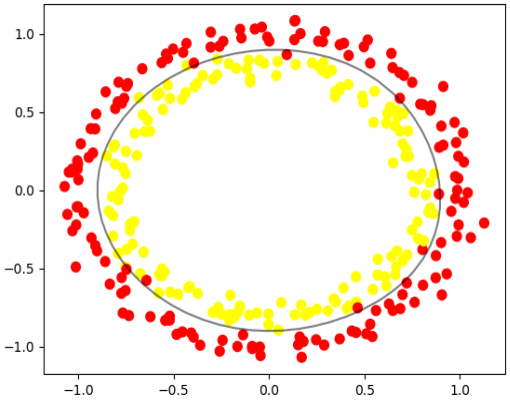


Figure.3-3: Binary SVM classification with non-linear decision boundary: RBF kernel [10]

The other solution for a non-linear separable dataset is mapping the data to a higher-dimensional space, in which the data is separated linearly (see Figure.3-4). This is known as the kernel trick [11].

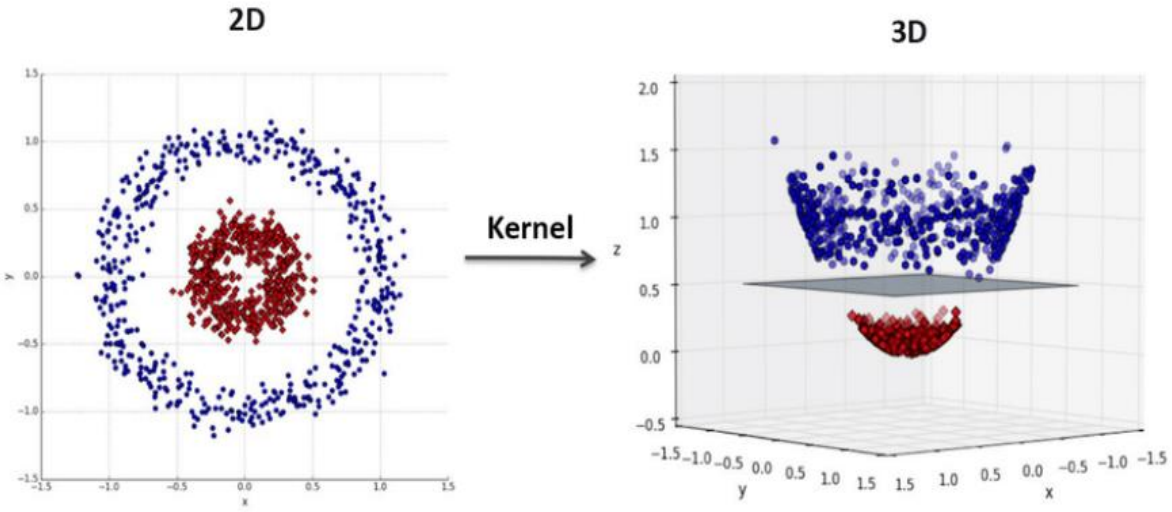


Figure.3-4:Non-linear SVM classification using kernel trick [11]

3.1.2 Multi classification SVM

SVM is primarily used for binary classification, and its extension to multi-class classification is an ongoing research issue [12]. The two most commonly used algorithms are one-vs-one and one-vs-all. The one-vs-all algorithm divides the multi-class classification module into *l* Binary SVM models where *l* is the number of classes [12]. Thus, it should be found *l* optimal hyperplanes, which each of them separates the data points for each class from the data points of other classes (see Figure.3-5). The one-vs-

one algorithm divides the multi-class classification module into $l.\frac{l-1}{2}$ binary SVM models [12]. Thus, it should be found $l.\frac{l-1}{2}$ hyperplanes which each of these hyperplanes separates the data points between every two classes (see Figure.3-6).

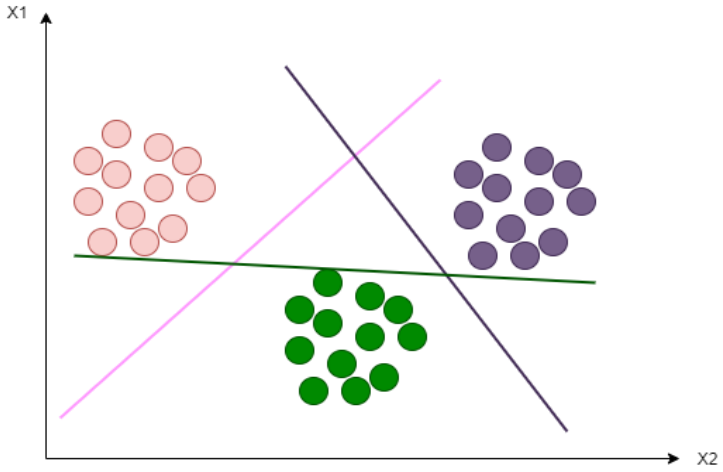


Figure.3-5: one-vs-all (based on [13])

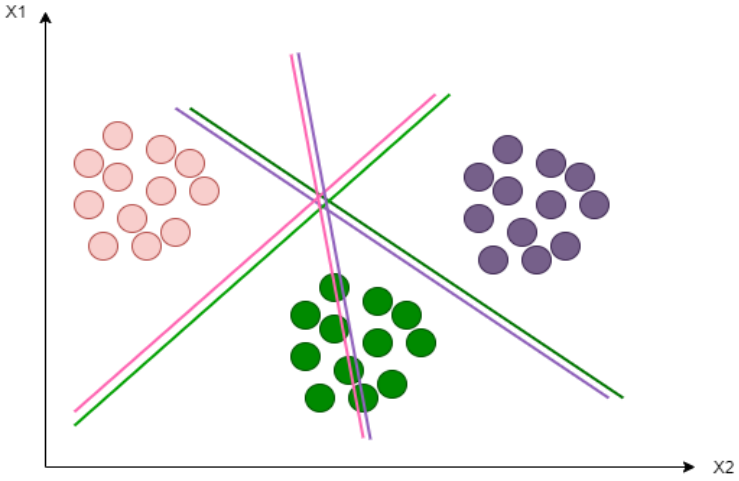


Figure.3-6: one-vs-one (based on [13])

3.2 K-Nearest Neighbor (KNN) classification

The principle of the KNN algorithm is simple, each data point is assigned the class most commonly found among its neighbors [14]. The number of neighboring points through which we want to determine the class of the data point is defined by the *k* parameter [15]. Let us assume that two classes are presented in two-dimensional space and we want to find out the class of the redpoint. Let's say that *k*=4 then that class will be the class of the four adjacent points inside a circle whose center is that redpoint (see Figure.3-7). In this case, the red point will belong to the green class. The KNN algorithm should not be used in the case of a large number of data points [15].

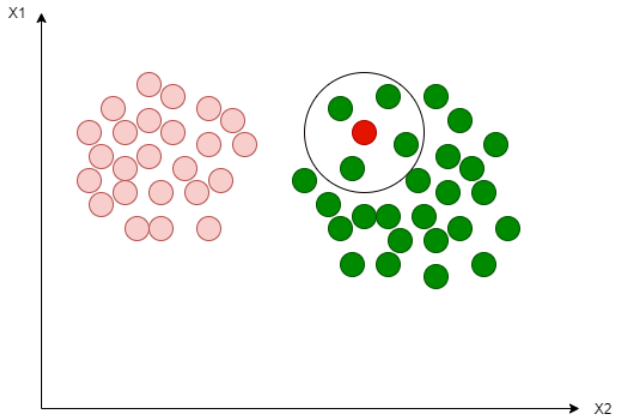


Figure.3-7: KNN classification

3.3 Neural Networks

Neural networks (NNs) imitate the mechanism of the human brain. As it is known, the brain is a network of neurons, where the association of these neurons forms specific information. The neural network is a network of thousands of nodes. The principle of processing the information by the node is as follows: the input values are multiplied by the weights before entering the node and then apply the results to a non-linear function (the activation function), mapping them to the output [16] (see Figure.3-8).

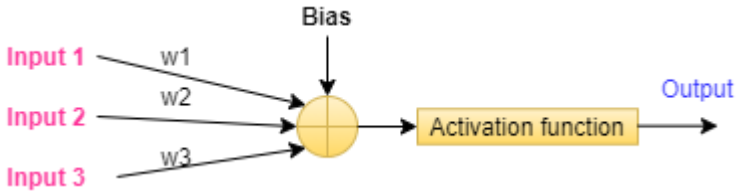
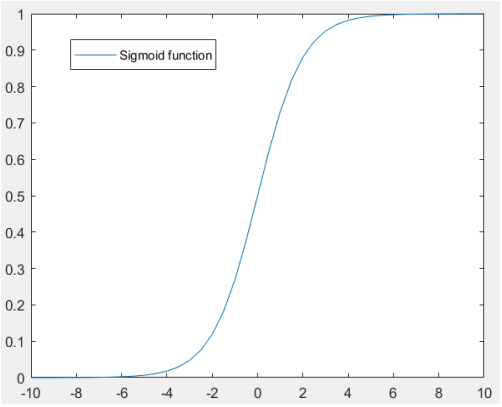
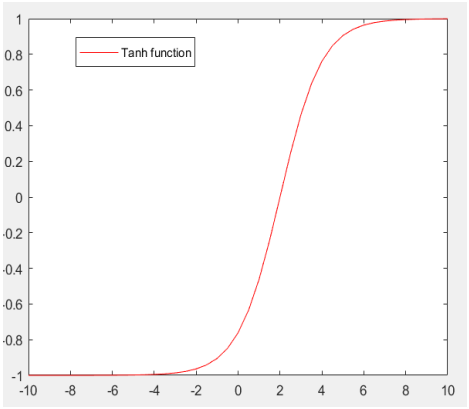


Figure.3-8: The neuron structure [15]

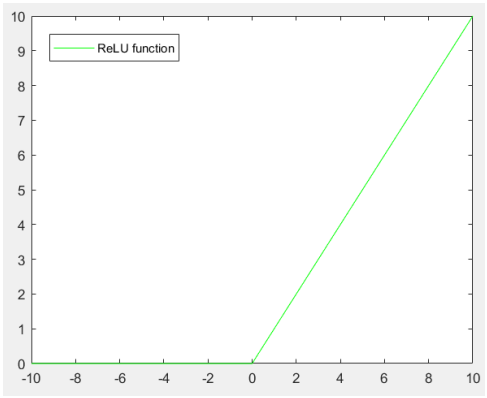
Popular neurons activation functions are: Sigmoid function, Tanh function and Rectified Linear Unit (ReLU) function [16] (see Figure.3-9).



(a)



(b)

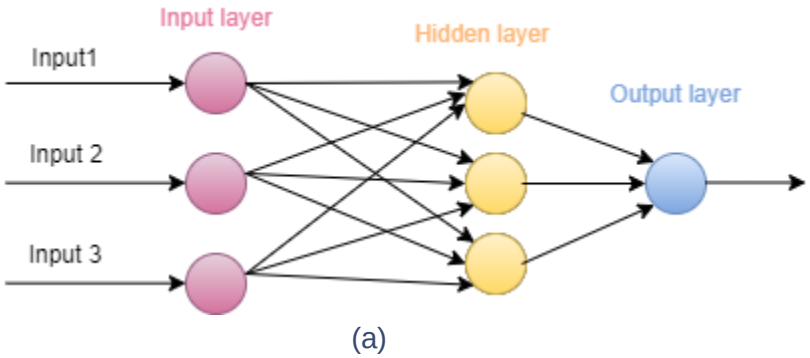


(c)

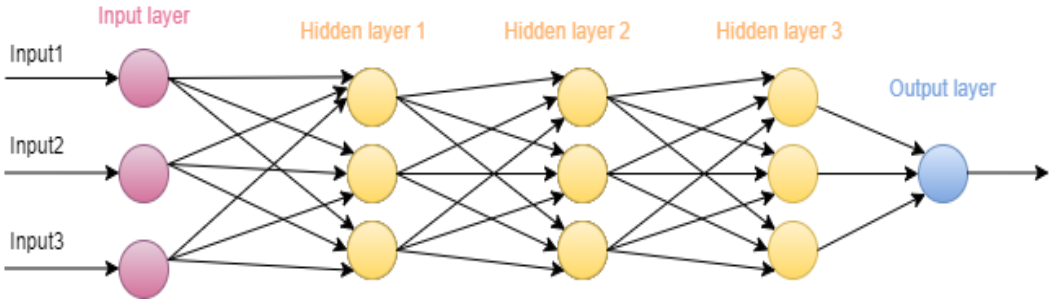
Figure.3-9: The activation functions: (a) Sigmoid function. (b) Tanh function. (c) ReLU function

The different types of NNs can be created by different types of connections among nodes. Most of today's neural networks used a layered connection where the neurons are connected to the sites by three layers: the input layer, one or multiple hidden layers, and the output layer [16]. According to the number of hidden layers, the neural networks can be divided into shallow neural networks and deep layer networks. The main difference between them is that Shallow NNs usually have only one hidden layer as opposed to deep NN, which has several hidden layers(see Figure .3-10) [17].

The principle of using neural networks for classifying data with the least possible error is as follows: the nodes of the input layer receive the input signal with the initialized input weighs and processing them by the nodes in the first hidden layer then transmitting them to the nodes of the second hidden layer and so on until they arrive at the output layer, where the classification decision is made [17]. This output decision is compared with the actual classification and if they do not match, then the network parameters are updated. The same procedure is repeated until the classification becomes accurate (see Fig.3-11)



(a)



(b)

Figure.3-10: Simplified diagram of Neural network : (a) shallow neural network. (b) Deep neural network

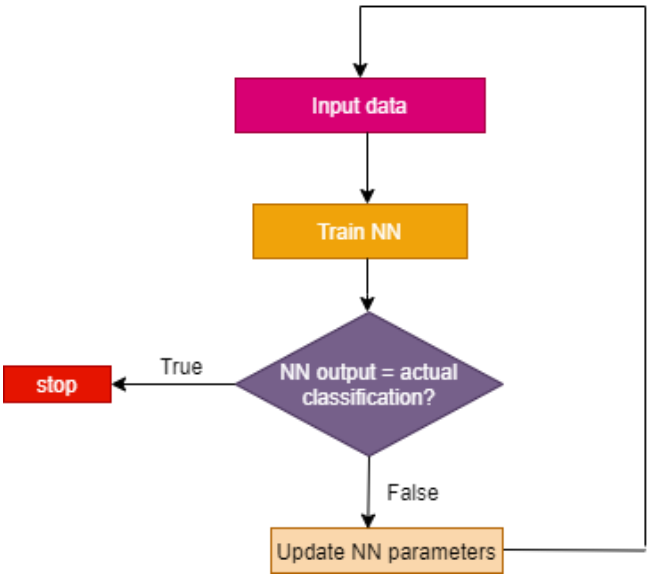


Figure.3-11: Simplified diagram of NN training algorithm

4. MEASUREMENT SETUP

As explained in the previous chapters, the main goal of the presented work is to classify the RF transmitters depending on their RF imperfections by using a selected machine learning algorithm. To do so, a laboratory workplace was established by Martin Pospisil from DREL, BUT [18]. This workplace consists of the PC that controls one of the universal software-defined radio (USRP) devices that emit an RF signal with specific features. The signal is received by a detector (antenna or direct cable connection) and Vector Signal Analyzer (VSA) Rohde&Schwarz FSVR with digital demodulation capability to estimate the RF signal's features suitable for wireless device authentication [18]. The features selection and their classification are done on the recognition system by Matlab software (see Figure.4-1).

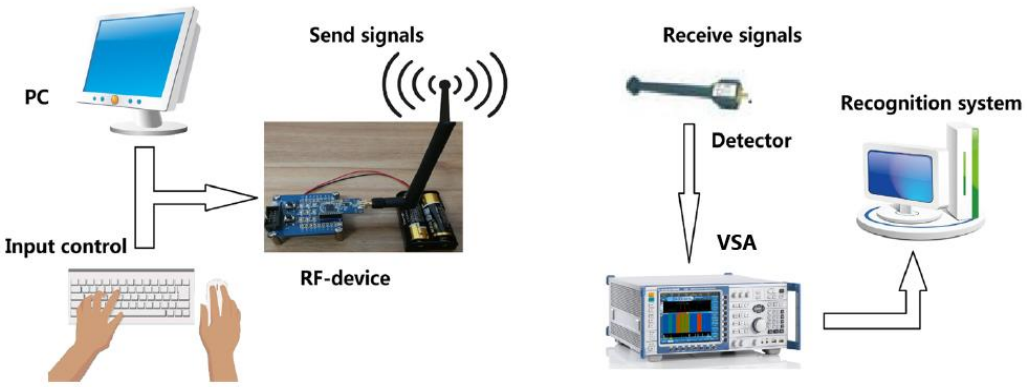


Figure.4-1: Laboratory workplace for sending and receiving the Rf signal [18]

A set of measurements was performed on a sample of eight different USRP N200 transmitters by Martin Pospišil [19]. As we know, the USRP radio consists of a mainboard and a front-end board. The combination of these boards in each transmitter is shown in Tab.4-1. The main parameters that were set for the transmitted signal and the VSA are summarized in Tab.4-2.

Tab.4-1: The combination of the mainboard and front-end board

	Mainboard	Mainboard serial number	Front-end name	Front-end serial number
RF1	N200 rev.2	E7R11YEUN	SBX rev.3.0	E8R11YCXS
RF2	N200 rev.2	E7R11YEUN	SBX rev.5.1	F4D019
RF3	N200 rev.2	E7R11YEUN	WBX rev.3.0	EAR23T4XW
RF4	N200 rev.2	E7R11YEUN	WBX rev.3.0	EDR1EV3XW
RF5	N200 rev.4	E6R14U6UN	WBX rev.3.1	EDR1EV3XW
RF6	N200 rev.3	EBR10ZAUN	WBX rev.3.0	E2R19X9XW
RF7	N200 rev.4	E5R14U6UN	WBX rev.3.1	EDR1EV1XW
RF8	N200 rev.4	E4R15TAUN	WBX rev.3.1	EDR1EV1XW

Tab.4-2: The main parameters that were set for the transmitted signal and the VSA

	Transmitted signal			
	Power [dBm]	**gain**	**Sample rate [MSps]**	**Type of modulation**
RF1	-1	10	1	QPSK
RF2	-1	16	1	QPSK
RF3	-1	24	1	QPSK
RF4	-1	24	1	QPSK
RF5	-1	24	1	QPSK
RF6	-1	24	1	QPSK
RF7	-1	24	1	QPSK
RF8	-1	24	1	QPSK

	VSA setting			
	Center frequency [MHz]	**Reference level [dBm]**	**Attenuator [dBm]**	**Symbol rate [kS/s]**
RF1	1800	-4.71	20	100
RF2	1800	4.2	20	100
RF3	1800	6.34	25	100
RF4	1800	7.07	25	100
RF5	1800	5.24	25	100
RF6	1800	5.24	25	100
RF7	1800	6.46	25	100
RF8	1800	5.26	25	100

5. IMPLEMENTING SVM AND NEURAL NETWORK WITH MATLAB

In this chapter, the practical side of the RF transmitters classification will be discussed, where SVM and neural network algorithms are used. The first and second sections explain some Matlab functions that are used to implement the SVM and Neural network algorithms. In the last section, the outputs of applying SVM for classification is discussed and compared with the outputs of applying neural network. The main steps for developing such an SVM and neural network algorithms by Matlab software are:

- Create a synthetic or use the measured dataset
- Split the dataset into training data and testing data
- Train the classifier with the training data
- Evaluate the algorithm by using the testing data.

5.1 The implementation of the SVM algorithm in Matlab

5.1.1 scenario 1: Build a simple SVM with artificial data

To illustrate how the SVM algorithm works, I tested first a simple case of a support vector machine classifier with artificial data corrupted by the white gaussian noise generated by the randn function. At first, the data were chosen to be linearly separable. Then a noise with varying dispersion was added to the data to investigate how the classifier performance varies as the dataset distribution changes. Some functions used in the code will be explained in the next section.

5.1.2 scenario 2: Build a SVM code with real data for binary classification

In this scenario, a support vector machine binary classifier is built in Matlab and tested with the example measured data of transmitters RF1 and RF4. The steps are as follows:

- **Step 1: Load the data**
  The measured dataset is stored in eight folders. You can select the file path on your computer to get to the folders.

- **Step 2: Select the features to be classified by the SVM algorithm for each transmitter**

  The features that can be selected are:
  CFO [Hz], gain imbalance [dB], IQ imbalance[dB], origin offset [dB], phase error [degree], quadrature error [degree]

Here, we have chosen only two features at once for each transmitter because the classification is binary and they will be presented in 2D plots.

- **Step3:Train an SVM classifier using the Linear kernel and the box constraint parameter if it is necessary:**

```
SVM=fitcsvm(Data,Class,'KernelFunction','linear',
'BoxConstraint',500);
```

In this case of binary classification, an SVM classifier is trained by the *fitcsvm* function. This function takes the data matrix and the classes matrix as inputs and returns a classifier SVM object, which is stored in *SVM* variable. The dataset in the *Data* variable where the CFO and the gain imbalance have been used as the SVM features is not entirely separable. For this reason, The *BoxConstraint* parameter is used. The *BoxConstraint* parameter presents the penalty coefficient *C* which is mentioned in chapter 3.1.1.3. *Boxconstraint* is equal to 1 by default. It has been noticed that the bigger the *C*, the fewer fewer support vectors the decision boundary will depend on. The best fit was obtained at *C* = 500.

- **Step4: getting the support vectors by using the parameter SupportVectors provided in the SVM module**

```
SupportV= SVM.SupportVectors;
```

- **Step5:Finding the parameters of the decision boundary**

```
LinearR= fitclinear(Data,Class);
```

There are several software functions for efficient formulation the decision boundary. I use the *fitclinear* function to find the parameters of the decision boundary. The *fitclinear* function takes the data matrix and the class matrix as inputs and returns a train linear classification module. It has been noticed that the *Beta* parameter and *Bias* parameter of the *fitclinear* module represent the parameters of the decision line.

- **Step6:Finding the equation of the decision boundary**

```
optimal= @(Data)-(LinearR.Beta(1)*Data+LinearR.Bias)/
LinearR.Beta(2);
```

As mentioned in chapter 3.1.1, the equation of the decision line is as follows:

$$w1.x1 + w2.x2 + b = 0 \implies x2 = \frac{-(w1.x1 + b)}{w2}$$

Here, LinearR.Beta(1) and LinearR.Beta(2) represent the *w1* and *w2* parameters. TheLinearR.Bias represents the *b* parameter.

- **Step7:Finding the Margin**

```
M1=(LinearR.Beta(1)*SupportV+LinearR.Beta(2)*..
SupportV+ LinearR.Bias)/ LinearR.Beta.*…
(LinearR.Beta)';
```

The margin is simulated according to the distance of the line from the support vector point. The Margin is given by this equation: $\frac{w1.x1+w2.x1+b}{\|w\|}$ where x_1 are the support vectors.

.

- **Step8: Get the classification error**

```
L=loss(SVM,Data,Class)
```

- **Step9: Receiver Operating Characteristic (ROC) curve**

```
SVMtest= fitcsvm(DataTest,ClassTest)
mdlSVM = fitPosterior(SVMtest);
[lable,pro_svm]= resubPredict(mdlSVM);
[Xsvm,Ysvm] = perfcurve(ClassTest,...
pro_svm (:,mdlSVM.ClassNames),'true');
```

ROC curve is used to evaluate the performance of the used SVM classifier where the false positive rate and the true positive rate were found by the *perfcurve* function. In our case (binary classification between two transmitters), the false-positive rate and the true positive rate can be explained byTab.5-1.

Tab.5-1: The true positive rate and the false positive rate

The real classification	***The output classification***	***Probability***
Transmitter1	*Transmitter1*	the true positive rate
Transmitter1	*Transmitter2*	the false-positive rate
Transmitter2	*Transmitter2*	the true-positive rate
Transmitter2	*Transmitter1*	the false-positive rate

5.1.3 scenario 3: Build a SVM code with real data for multi-class classification

In this scenario, a support vector machine multi-class classifier is built in Matlab with the measured data of the transmitters. The steps are as follows:

- **Step1: Select the features to be classified by the SVM algorithm for each transmitter**
  During the first experiments, two features were chosen. In the subsequent experiments we have used three features to know how the classifier

performance varies as the number of used features changes.

- **Step2: Create a template for a binary SVM using the function *templateSVM***

```
template = templateSVM( 'Standardize',1);
```

- **Step3: Train an SVM classifier for multi-class classification**

```
SVMMul=fitcecoc(Data,Class,'Learners',template);
```

In the case of multi-class classification, the SVM classifier is trained by the *fitcecoc* function. This function takes the data matrix,and the classes matrix as inputs and returns error-correcting output codes (ECOC) classifier which is stored in *SVMMul* variable.

- **Step4: Get the decision boundary parameters and find the appropriate function**

```
for i = 1:length(SVMMul.BinaryLearners)
svm = SVMMul.BinaryLearners{i};
%in the case of 2 features
x1vals = linspace(min(DataAll(:,1)),max(DataAll(:,1)));
func =@(x1)-svm.Beta(1)/svm.Beta(2) * ((x1-
svm.Mu(1))./svm.Sigma(1)) - svm.Bias/svm.Beta(2);
x2vals = func(x1vals)*svm.Sigma(2)+.. svm.Mu(2);
hold on;
end
```

This script is written according to the one-vs-one algorithm, as mentioned in chapter 3.1.2. BinaryLearners is an SVMMul property to get the parameters for the decision boundary where each binary learner works with two classes. The decision boundary can be a line (in the case of choosing two features for classification) or a hyperplane (in the case of choosing three features). The following script applies to the first case.

- **Step5: Greate the confusion Matrix to evaluate the performance of the used SVM classifier**

```
plotconfusion(Truelabels,PredictLabels);
```

This function plots a confusion matrix for the true labels *Truelabels* and predicted labels *PredictLabels*.

5.2 The implementation of Neural network algorithm in Matlab

This section introduces a brief description of some neural network Matlab functions used for classification:

- **Determine the number of neurons in the hidden layer of the neural network classifier**

  ```
  net= patternnet(20);
  ```

  Here, the number of neurons in the hidden layer is equal to 20.

- **Train a neural network classifier**

  ```
  net = train(net,DataAll,Class);
  ```

- **Classify data and evaluate the performance of the neural network classifier**

  ```
  y= net(Data);
  ```

  The net function takes the data matrix as input and returns the predicted label of the data where it is stored in the *y* variable.

  ```
  plotconfusion(tru,pre);
  ```

  this function plots a confusion matrix for the true labels *tru* and predicted labels *pre*.

5.3 Results

The following section describes the outputs of applying the classification by SVM and neural network algorithms in Matlab.

The SVM algorithm is discussed and its performance is shown in Figure.5-1 to Figure.5-17. In these figures, the points enclosed in the yellow circles are the support vectors we have found.

From Figure.5-1, It is evident that the data are linearly separable. In this case. the SVM classifier best separates the data into two classes and thus there will be no classification errors.

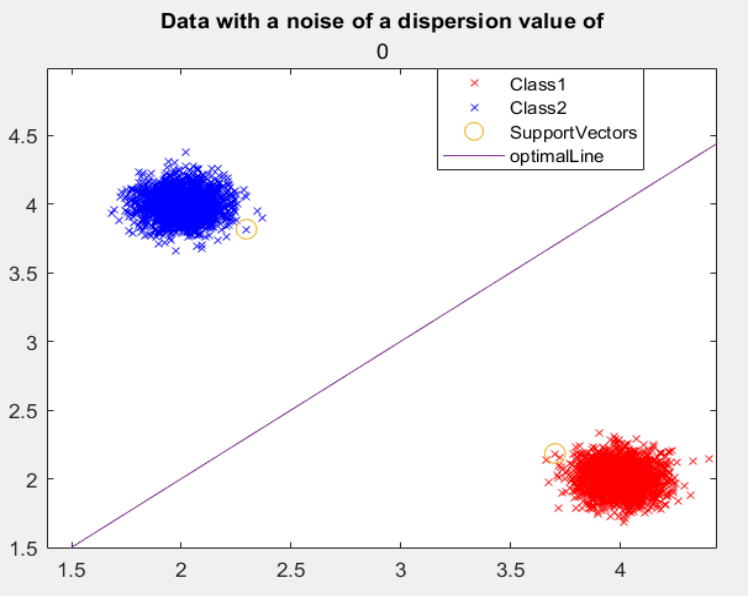


Figure.5-1: SVM binary classification with artificial data(linear separable case)

In Figure.5-2, a white gaussian noise signal with the dispersion of 0.7 is added to the data. The dataset then becomes only nonlinearly separable. This case introduces a soft margin SVM where some amount of data points are strayed over the line into the margin. The line becomes overfitting, resulting in increasing the classification error.

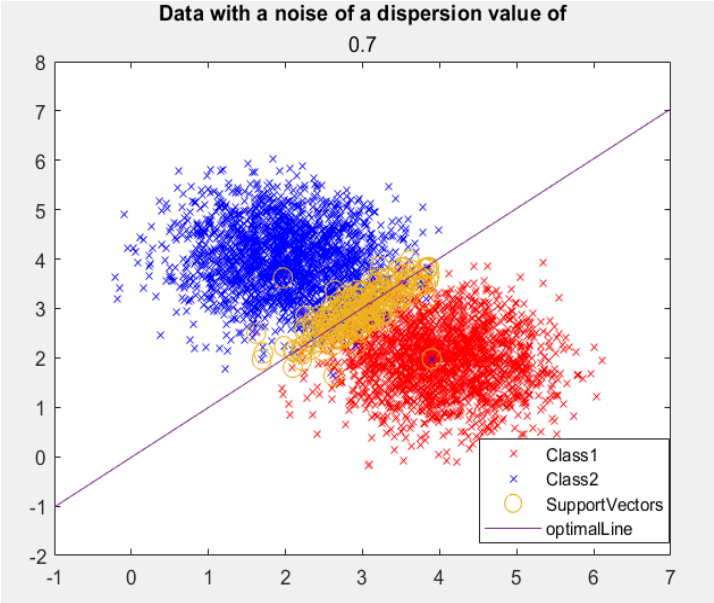


Figure.5-2: SVM binary classification with artificial data (linear non-separable case)

Figure.5-3 presents the ROC curve to evaluate the performance of the used SVM classifier. It can be seen, that the performance of the classifier is degraded with increased noise dispersion.

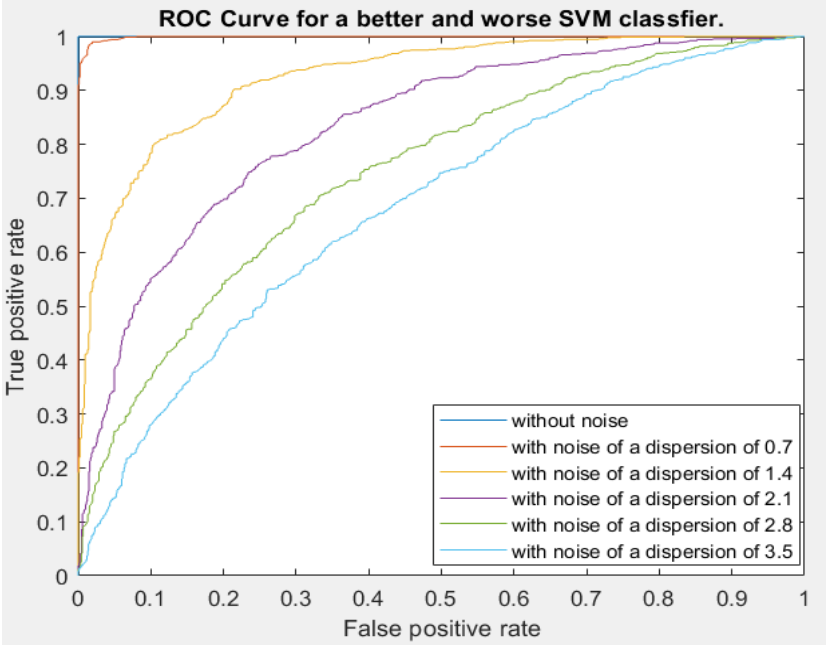


Figure.5-3: ROC curve for better and worse SVM binary classifier with artificial data

In Figure.5-4, the situation of SVM classification based on gain imbalance and quadrature error for transmitters RF1 and RF4 is shown. It is evident that these features are linearly separable and thus the SVM classifier works ideally.

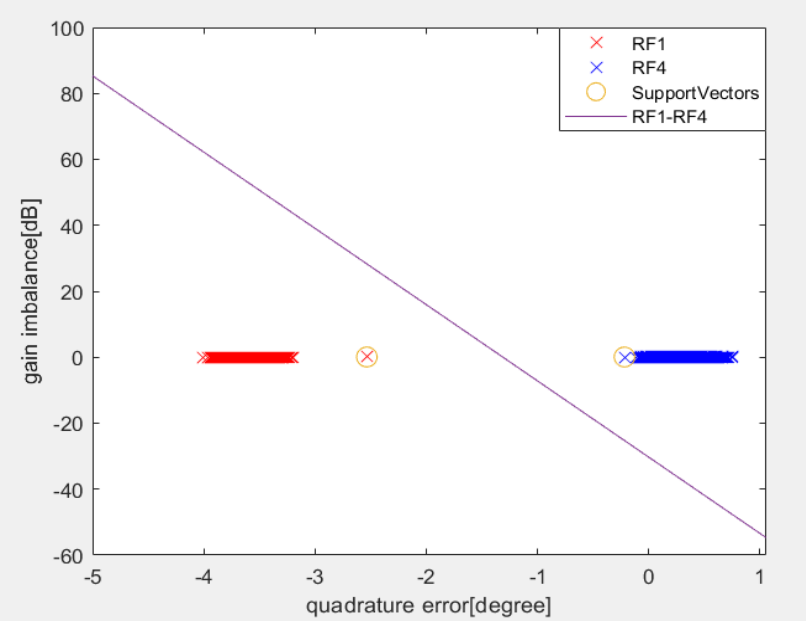


Figure.5-4: Binary classification for measured data of RF1 and RF4 (linear separable case)

Figure.5-5 presents the results of SVM classification based on the gain imbalance and the CFO. It can be noticed that the number of support vectors is increased when the features become non-linearly separable. Therefore, slack parameter *C* was used to control

the overfitting. From the figure, it is possible to see that over-fitting still occurs even though the *C* parameter was tuned. The reason for this is the used linear kernel function. This problem can be solved by using an additional feature for classification where the data becomes linearly separable. In this case, the added feature is quadrature error (see Figure.5-6).

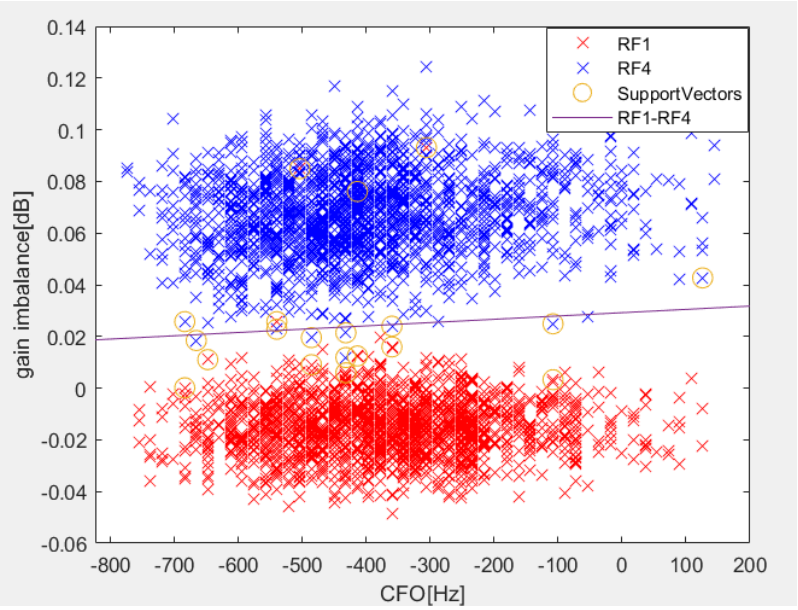


Figure.5-5: Binary classification for measured data of RF1 and RF4. The CFO and the gain imbalance have been used as the SVM features

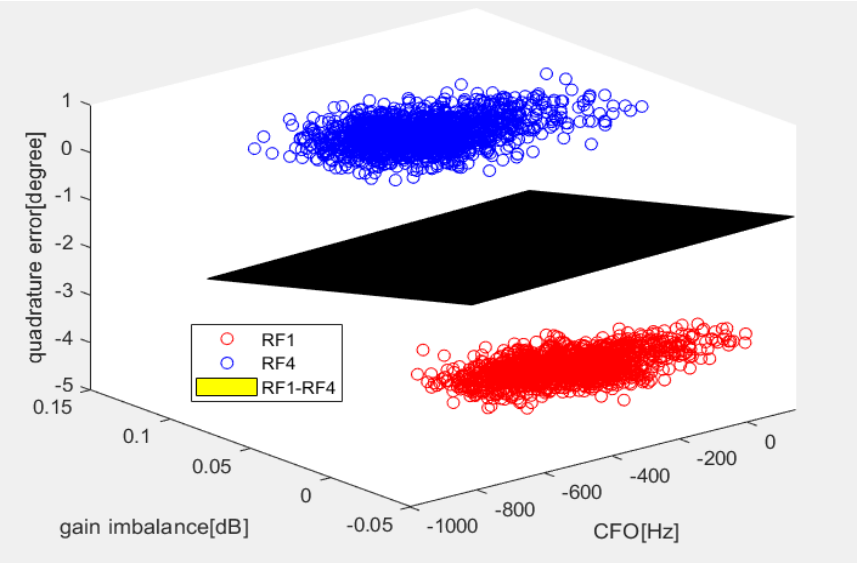
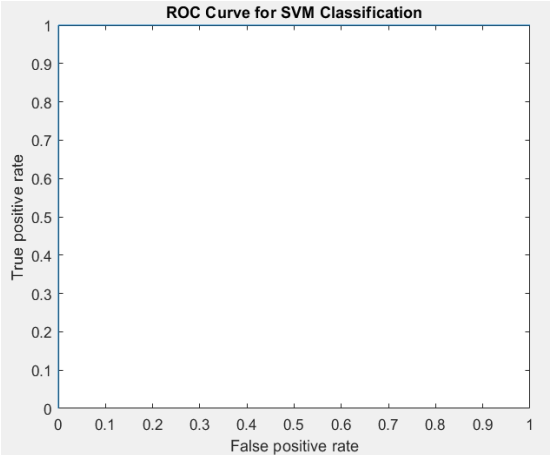
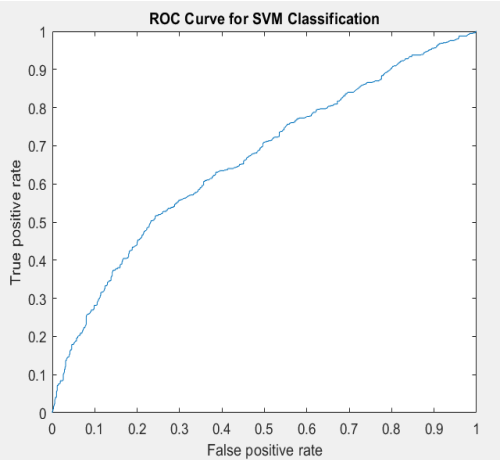


Figure.5-6: Binary classification for measured data of RF1 and RF4. The CFO, the gain imbalance, and the quadrature error have been used as the SVM features

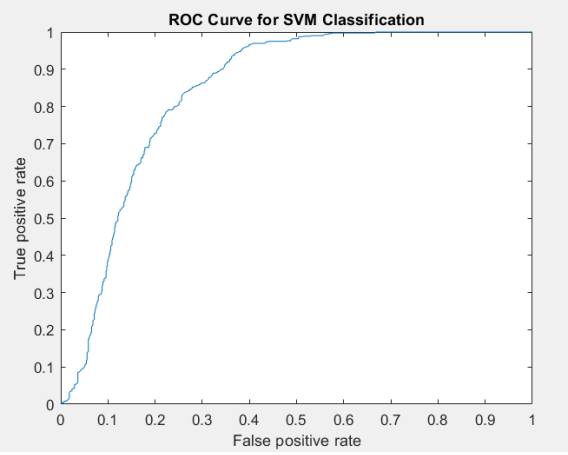
Figure.5-7, Figure.5-8 present various ROC curves and confusion matrix for binary classification between transmitter RF1 and transmitter RF4, transmitter RF3 and transmitter RF4, and transmitter RF1 and transmitter RF2 where the CFO and the gain imbalance have been used as the SVM features. The ROC curves and the confusion matrix are used to evaluate the performance of the SVM classifier. Each row in the confusion matrix corresponds to the predicted class and each column corresponds to the true class. It is possible to see that the best performance of the SVM classifier is obtained when classifying between RF1 and RF4 transmitters where the classifier works ideally because the selected data for evaluating the SVM classifier were almost linearly separable (the confusion matrix shows that the total classification accuracy is equal to 99.9%). The worst performance of the SVM classifier is obtained when classifying between transmitter RF3 and transmitter RF4 because of the high data overlapping between these transmitters (the confusion matrix shows that the total classification accuracy is equal only to 61%).



(a)

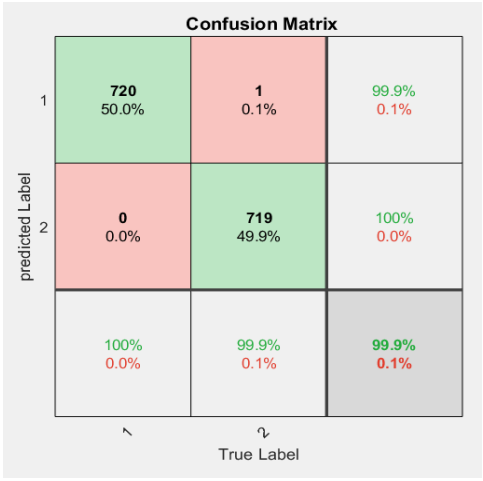


(b)

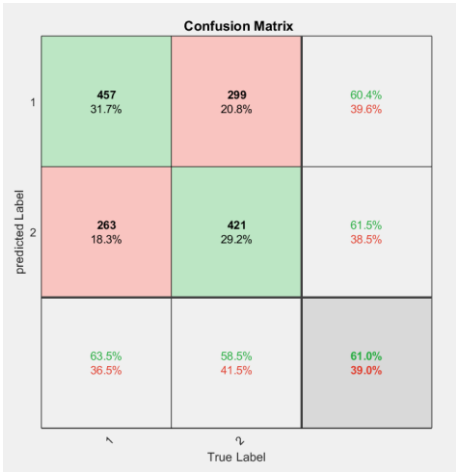


(c)

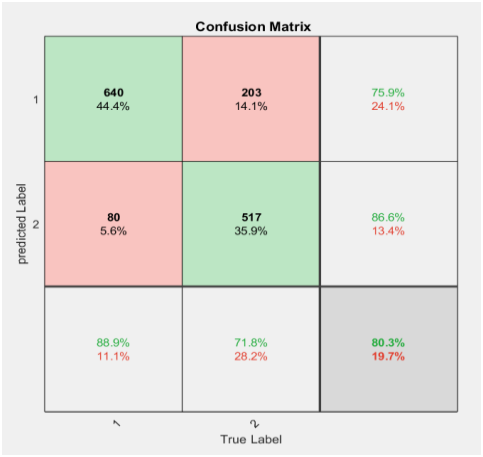
Figure.5-7: ROC curves for binary SVM classification: (a) between RF1- RF4. (b) between RF3- RF4. (c) between RF1- RF2



(a)



(b)



(c)

Figure.5-8: Confusion matrix for binary SVM classification: (a) between RF1- RF4. (b) between RF3- RF4. (c) between RF1- RF2

Figure.5-9, illustrates the multi-class SVM classification according to the one-vs-one algorithm. The first, second, and third-class present the measured data of transmitters RF1, RF4, and RF7, respectively. At first, the classification is done based on the gain imbalance and the CFO features then it is done based on the gain imbalance, CFO, and quadrature error features. From the figures, it can be seen that the yellow line (or hyperplane) separates the data of the RF1 from the data of the RF4 while the red line (or hyperplane) separates the data of the RF1 from the data of the RF7. Finally, the green line (or hyperplane) separates the data of the RF4 from the data of the RF7.

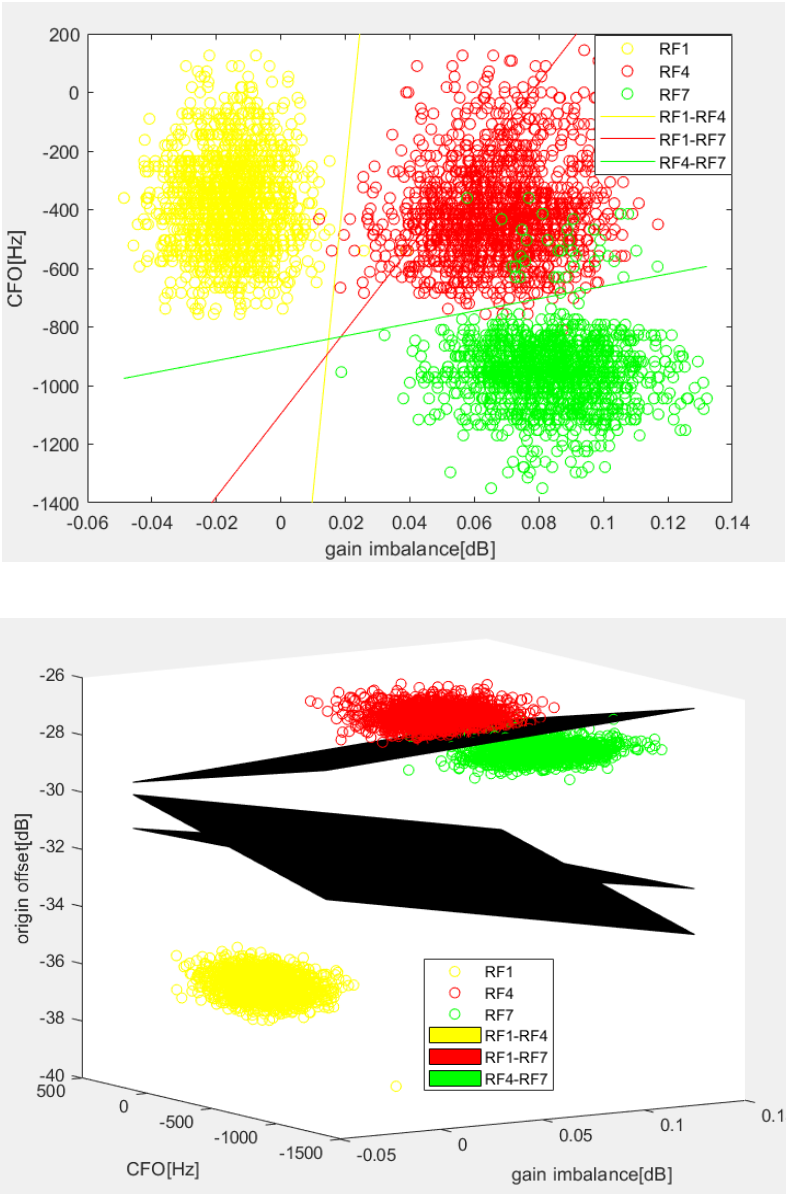


Figure.5-9: Multi-class classification for real data of RF1, RF4, and RF7.

Figure.5-10 presents the results of multiclass SVM classification based on the gain imbalance and the CFO where eight transmitters are used. The eight classes present the measured data of transmitters RF1, RF2, RF3, RF4, RF5, RF6, RF7, and RF8, respectively. The confusion matrix has eight rows and eight columns because the number of used transmitters is eight. Each row corresponds to a predicted class and each column corresponds to a true class. 720 observations from each of the eight transmitters were tested.

The interpretation of SVM performance for the confusion matrix in Figure.5-10 is not an easy task and may be misleading. The emergence of such results can be explained by the nature of the One-vs-One (OAO) multiclass algorithm as follows:

As mentioned in chapter 3.1.2, the multiclass classification problem according to the OAO algorithm is decomposed into $l.\frac{(l-1)}{2}$ binary classifiers, where l is the number of classes in the multiclass problem. In our case, the number of classes is equal to eight and thus the binary learners is equal to 28. The classes that each binary classifier works with are listed in Tab.5-2 where each column corresponds to a binary classifier for a pair of classes i and j.

In our case, 720 observations for each of the 8 transmitter classes were processed in (l-1=7) binary classifiers. Thus each transmitter can be identified in up to 5040 occurrences. In the case of correct detection, all these 5040 occurrences will appear on the main diagonal of the confusion matrix.

Moreover, the non-zero off-diagonal elements do not necessarily reflect the poor classifier performance, because, for each of 8 transmitter classes, there are 21 binary classifiers not considering the true transmitter class, resulting in (720.21= 15120) off-diagonal occurrences even for correct detection. As consequence, the total classification accuracy is disturbed [20].

Tab.5-2: The classes that each binary classifier works with

Learner L	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Class i	1	1	1	1	1	1	1	2	2	2	2	2	2	3
Class j	2	3	4	5	6	7	8	3	4	5	6	7	8	4

Learner L	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Class i	3	3	3	3	4	4	4	4	5	5	5	6	6	7
Class j	5	6	7	8	5	6	7	8	6	7	8	7	8	8

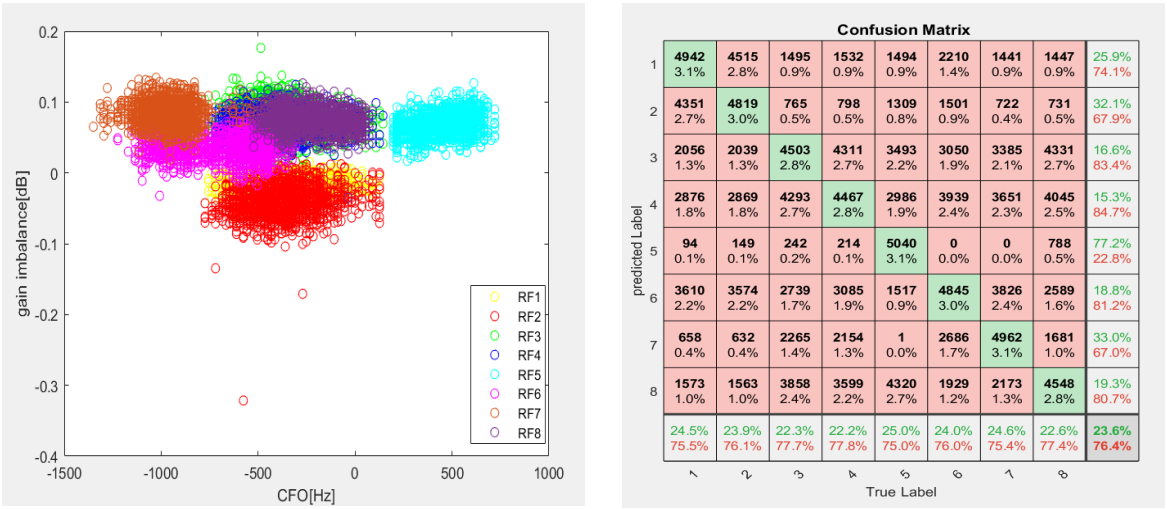


Figure.5-10: The data distribution for the eight transmitters and their confusion matrix. The CFO and the gain imbalance have been used as the SVM features

Figure.5-11, presents the results of SVM classification based on the CFO and the quadrature error. Visible differences can be seen in the data distribution of the RF3, RF4, RF5, RF6, RF7, and RF8 transmitters. There is still high data overlapping between RF1 and RF2. The explanation of the confusion matrix is similar to the explanation that was introduced for Figure.5-10.

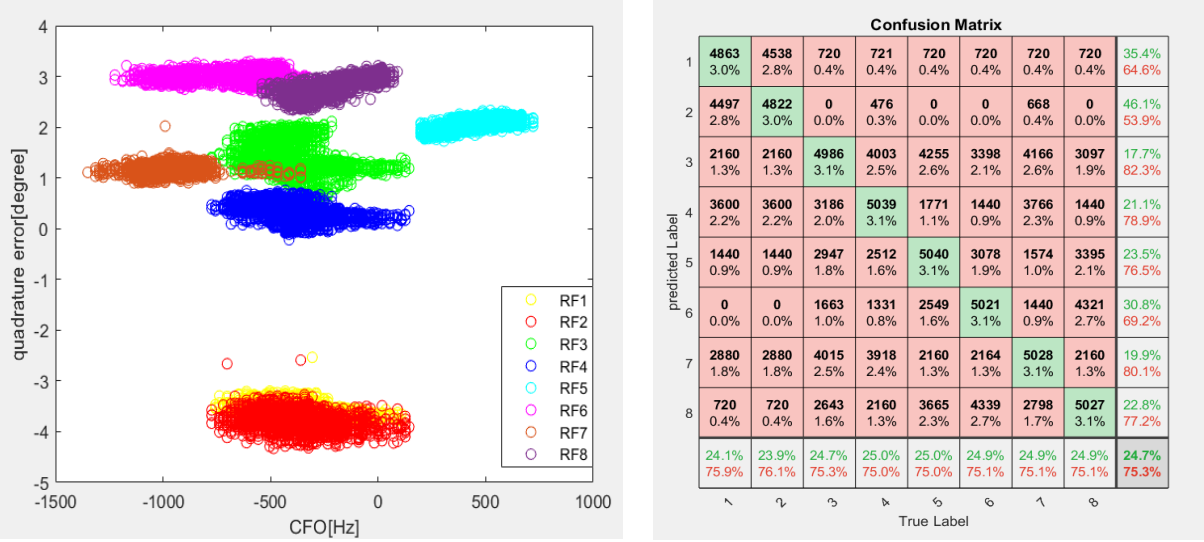


Figure.5-11: The data distribution for the eight transmitters and their confusion matrix. The CFO and the quadrature error have been used as the SVM features

Figure.5-12 presents the results of multi-class classification based on the origin offset and the CFO. The data became better separable after replacing the quadrature error with the origin offset.

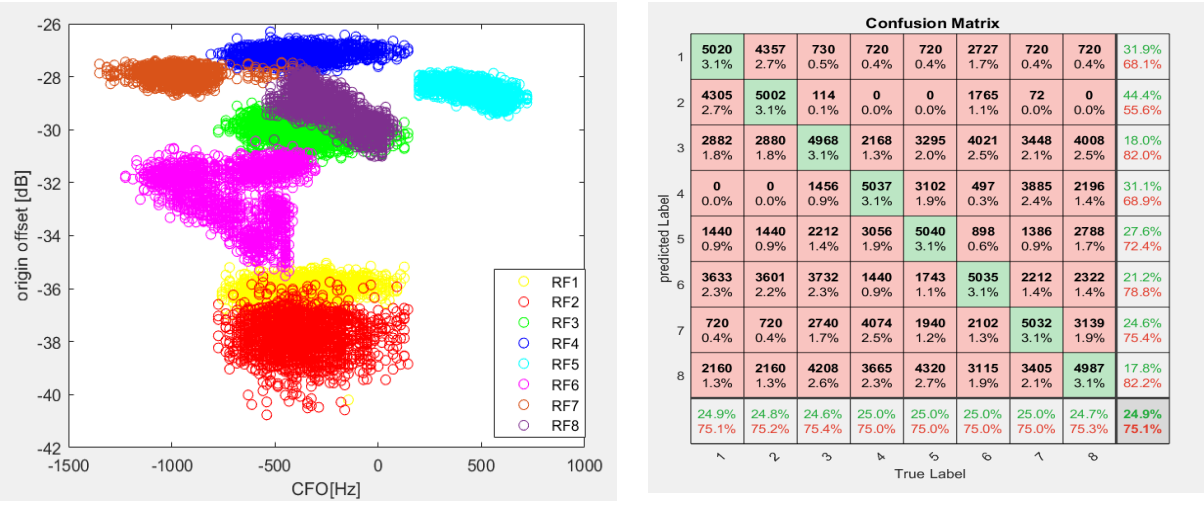
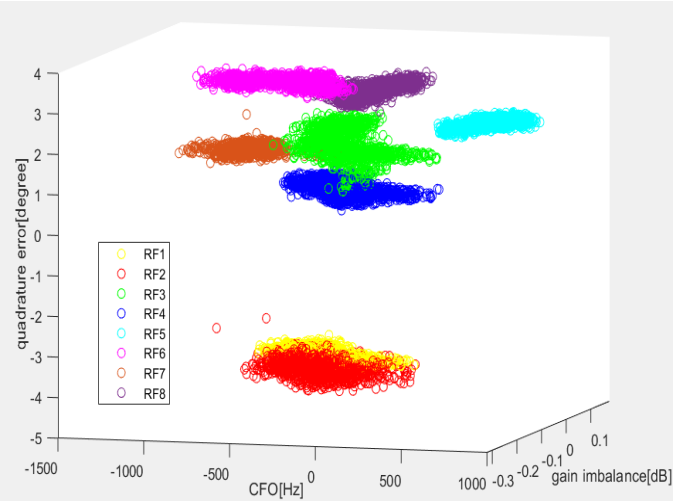


Figure.5-12: The data distribution for the eight transmitters and their confusion matrix. The CFO and the origin offset have been used as the SVM features

Figure.5-13 presents the results of SVM classification based on the gain imbalance, CFO, and quadrature error.



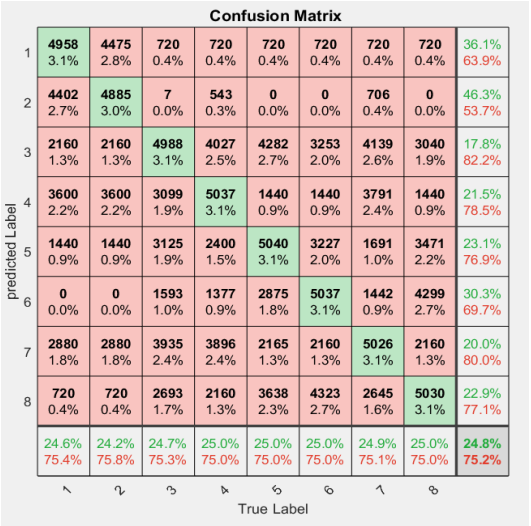
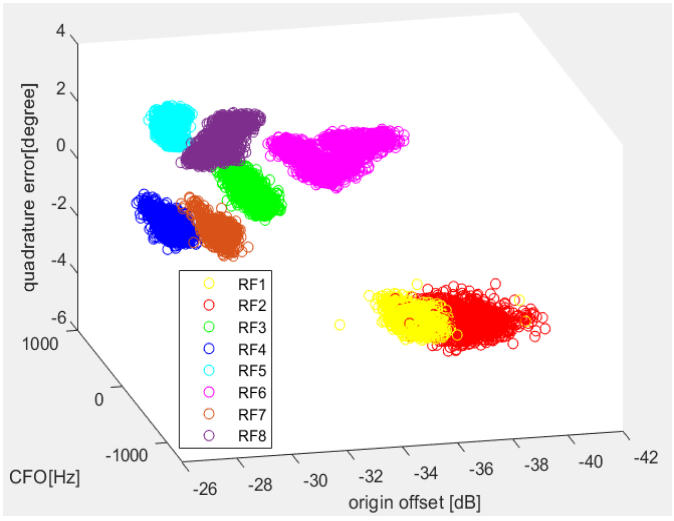


Figure.5-13: The data distribution for the eight transmitters and their confusion matrix. The gain imbalance, CFO, and the quadrature error have been used as the SVM features

Figure.5-14 presents the results of SVM classification based on the origin offset, CFO, and quadrature error. The dataset is highly separable. It can be noticed that all main diagonal of the confusion matrix has almost 5040 occurrences.



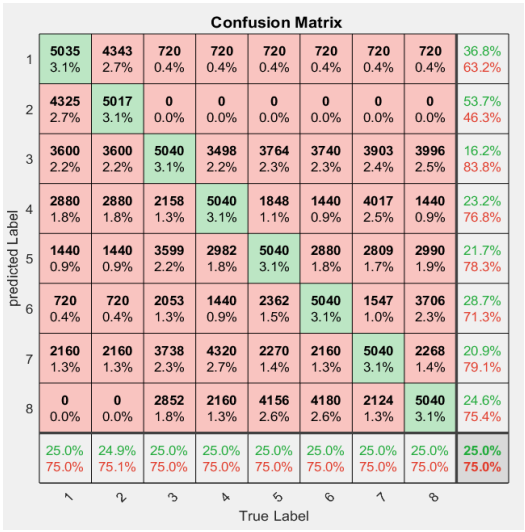


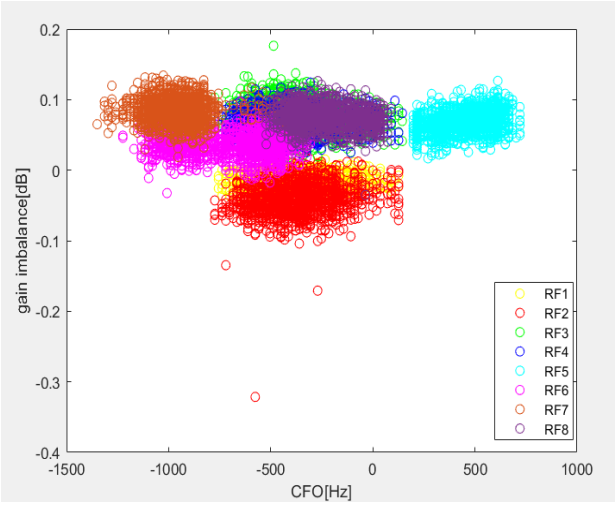
Figure.5-14: The data distribution for the eight transmitters and their confusion matrix. The origin offset, CFO, and the quadrature error have been used as the SVM features

Figure.5-15, Figure.5-16 presents the confusion matrix according to the majority voting strategy called ‘MaxWins [21]. 720 observations from each of the eight transmitters are segmented into 10 segments of 72 samples. The principle of constructing this confusion matrix is as follows: for each segment, all binary classifiers assign the test data into its preferred class. Then, the outputs of all binary classifiers are combined to get the multiclass prediction such as the predicted class will be the class with the maximum probability of occurrences (see Figure.5-17).

The elements of the confusion matrix represent the probability of prediction for j-th class (rows) in the case of transmission by i-th transmitter (column).

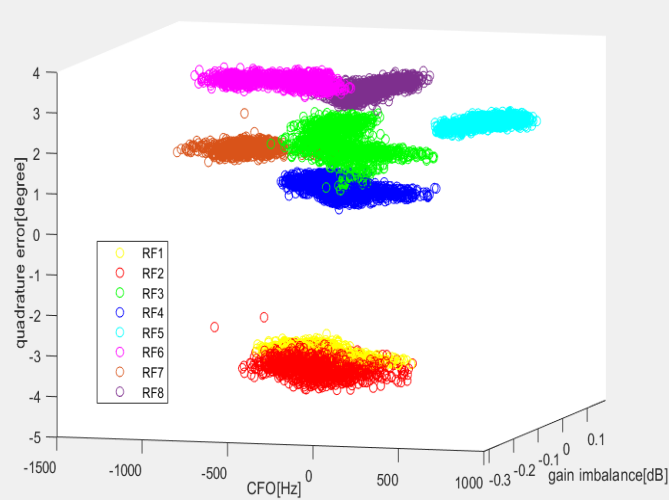
Let us discuss the results in the first column, where the true class is class 1. It can be noticed that the SVM classifier seems to work ideally in the case of classification between RF1 and RF2 transmitters, although the data of these transmitters are highly overlapped. The fact is that the probability of occurrence class 2 data points is very close to the probability of occurrence class 1 data points, but according to the MaxWins strategy, the winning class will be the one with the highest probability, which is class 1.

Note that in this particular case, the MaxWins strategy performed well, but this does not need to hold in general, in particular when the probability of occurrence of several classes is equal or very close to each other.



1	0.2	0	0	0	0	0	0
0	0.8	0	0	0	0	0	0
0	0	0.5	0.2	0	0	0	0.4
0	0	0.4	0.6	0	0.1	0	0
0	0	0	0	1	0	0	0
0	0	0	0	0	0.9	0	0
0	0	0	0	0	0	1	0
0	0	0.1	0.2	0	0	0	0.6

Figure.5-15: The data distribution for the eight transmitters and their confusion matrix. The gain imbalance and CFO have been used as the SVM features



1	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0
0	0	0.9	0	0	0	0	0
0	0	0.1	1	0	0	0	0
0	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1

Figure.5-16: The data distribution for the eight transmitters and their confusion matrix. The gain imbalance, CFO, and quadrature error have been used as the SVM features

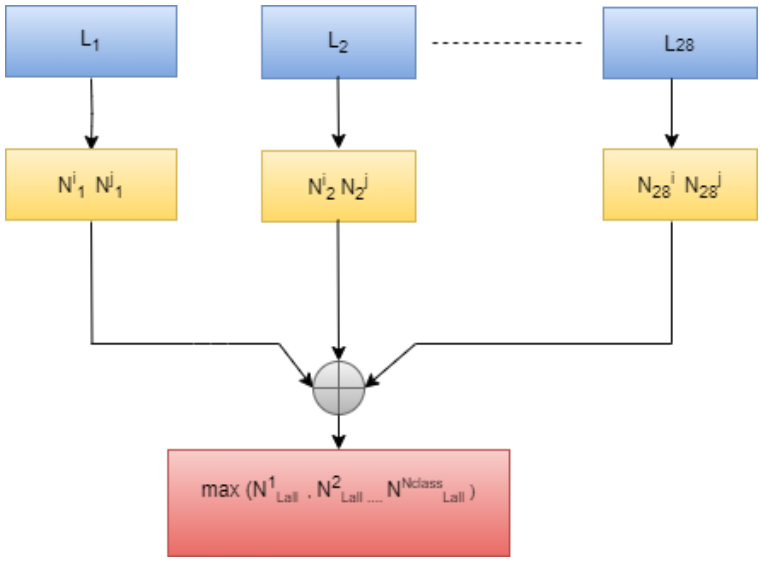


Figure.5-17: Simplified diagram of MaxWins strategy. The red block selects such a transmitter having the maximum number of occurrences accumulated overall binary learners NLall

the outputs of applying the classification by neural network algorithm are discussed as shown in Fig. 5-18 to Figure.5-24.

In Figure.5-18, a single hidden layered neural network with varying number of neurons {1, 20, 40, 60, 80, 100, 120, 140, 160, 180} was tested to know how the neural network classifier performance varies as the number of the used neurons changes. Note that a very good performance was obtained for the number of neurons equal to 20. It can be noticed that using number of neurons greater than 20 does not affect the performance of the classifier.

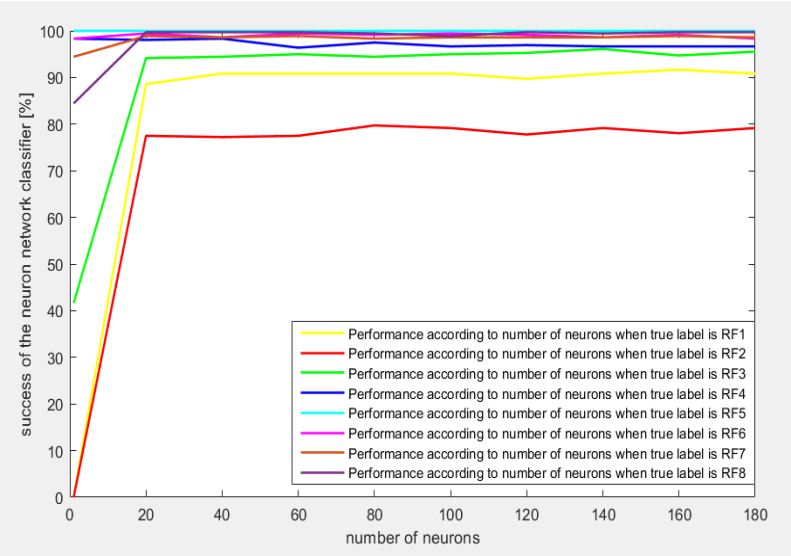


Figure.5-18: Neural network performance as a function of the number of neurons in the hidden layer

Now, the optimal number of hidden layer neurons has been determined (20 neurons). To get the results shown in Figure.5-19 to Figure.5-24, the neural network classifier was trained with 2 or 3 input neurons (the number of used features), 20 neurons in the hidden layer, and output nodes equal to the number of classes (eight classes).

Figure.5-19, presents the results of neural network classification based on the gain imbalance and the CFO. The confusion matrix has eight rows and eight columns because the number of used transmitters is eight. Each row corresponds to a predicted class and each column corresponds to a true class. 360 observations from each of the eight transmitters are tested.

From the confusion matrix, it is possible to see that the highest neural network performance degradation was observed when classifying between RF3-RF4 transmitters (60 to 70% observations misclassified) while the best performance was observed when classifying RF5 transmitter (100 % correctly classified). Significant errors also occurred in the classification between RF1-RF2. The reason for this is the high data overlapping between RF1-RF2, RF3-RF4-RF8 transmitters. Anyway, the total neural network classification accuracy achieves 72.6%.

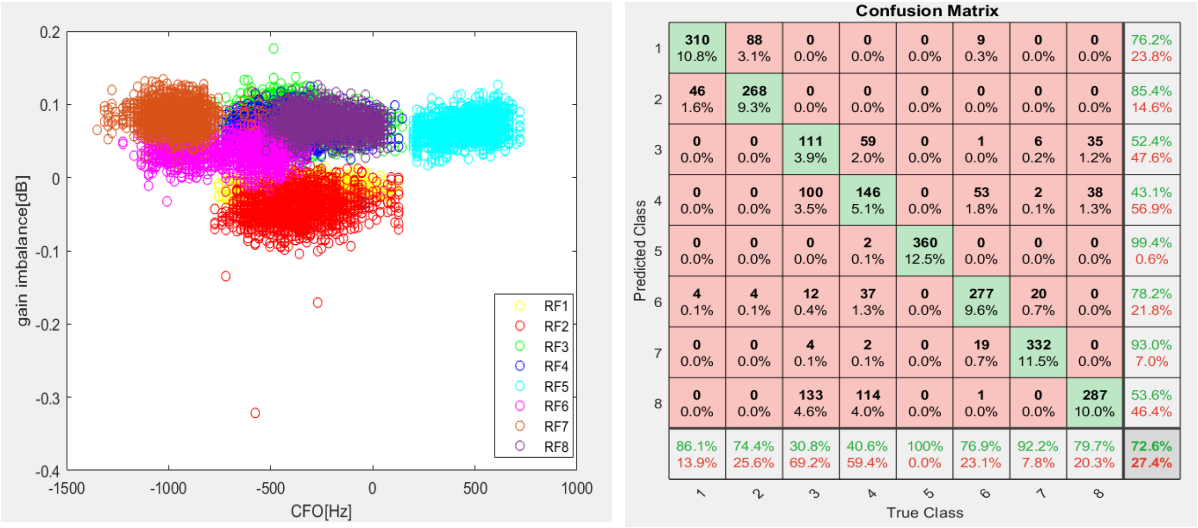


Figure.5-19: The data distribution for the eight transmitters and their confusion matrix. The CFO and the gain imbalance have been used as the neural network features

Figure.5-20, presents the results of neural network classification based on the CFO and the quadrature error. Visible differences can be seen in the data distribution of the RF3, RF4, RF5, RF6, RF7, and RF8 transmitters where the data of these transmitters became better separable after replacing the gain imbalance with the quadrature error. The data overlapping between RF1-RF2 transmitters is increased. The reason for this can be the selected features. Selecting other combinations of features may decrease this overlapping between RF1-RF2 transmitters. Anyway, the quadrature error can play a significant role in the classification between the transmitters where the total neural network classification accuracy achieves 91.4%.

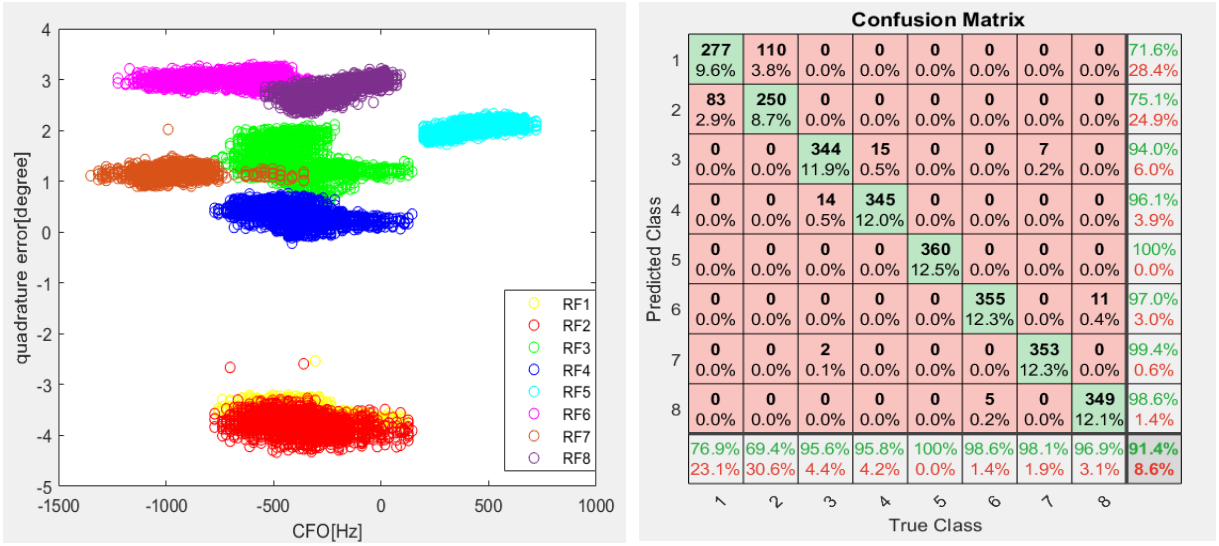


Figure.5-20: The data distribution for the eight transmitters and their confusion matrix. The CFO and the quadrature error have been used as the neural network features

Figure.5-21 presents the results of neural network classification based on the origin offset and the CFO. The data distribution of the eight transmitters became better separable after replacing the quadrature error with the origin offset. The confusion matrix shows that the neural network classification accuracy for each class has high values (at least 88.3%). The total neural network classification accuracy is 96.8%!.

That is mean the origin offset feature plays the most important role in the classification between all transmitters.

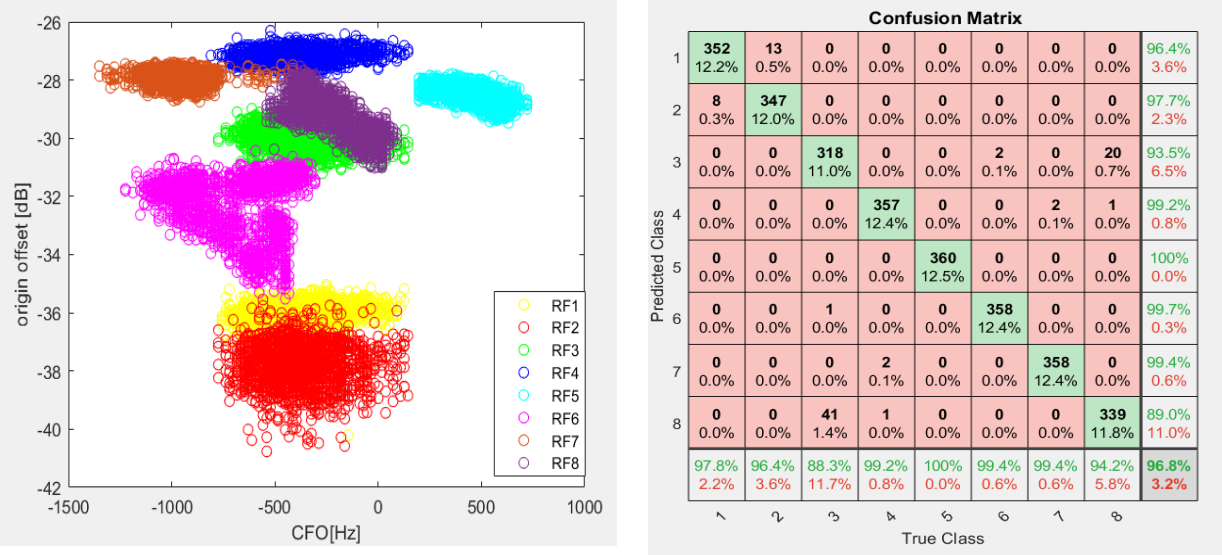


Figure.5-21: The data distribution for the eight transmitters and their confusion matrix. The CFO and the origin offset have been used as the Neural network features

Figure.5-22 presents the results of neural network classification based on the gain imbalance, CFO, and quadrature error. Compared to Figur.5-19, where only gain imbalance and CFO were used for classification, the data of the eight transmitters became better separated from each other, and we can see that the total classification accuracy became 94.7% while it was 72.6%. This means the distribution of the data in three-dimensional space makes the classification more robust. However, there still significant errors when classifying between RF1- RF2. The reason for this can be the selected features. Selecting other combinations of features may decrease the overlapping.

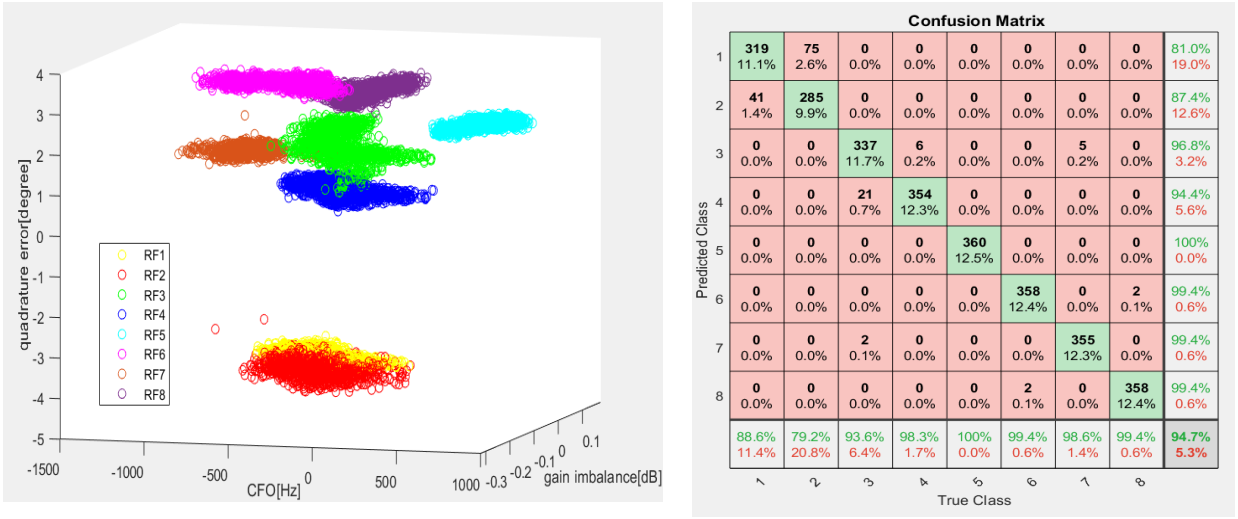


Figure.5-22: The data distribution for the eight transmitters and their confusion matrix. The CFO, gain imbalance and quadrature error have been used as the Neural network features

Figure.5-23 presents the results of neural network classification based on the origin offset, CFO, and quadrature error. It can be noticed that the data overlapping between RF1-RF2 transmitters noticeably decreased. The data of the other classes are almost totally separable. The total classification accuracy achieves 99.5% !.

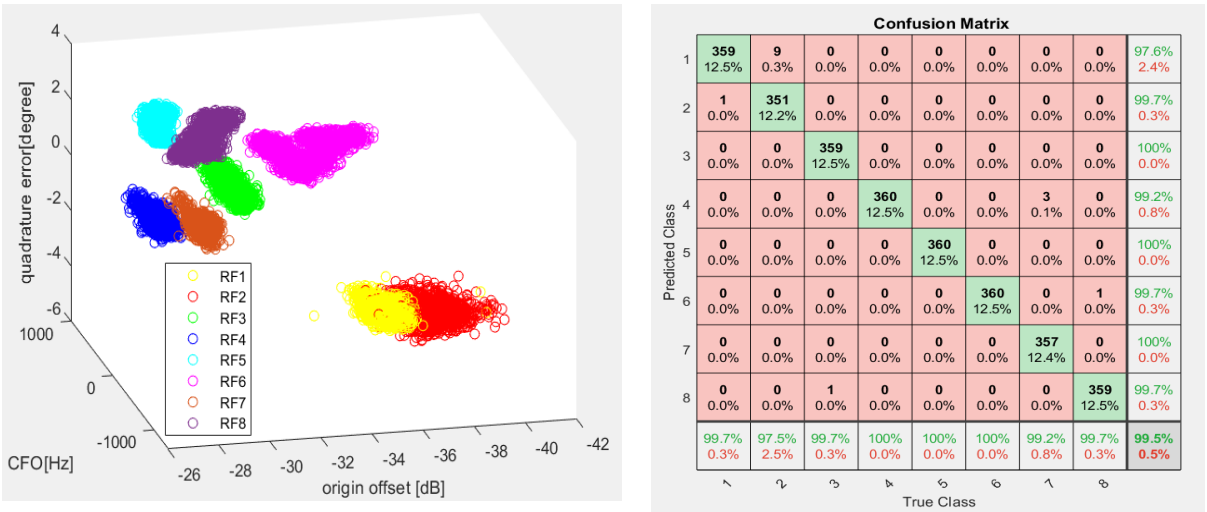


Figure.5-23: The data distribution for the eight transmitters and their confusion matrix. The CFO, origin offset, and quadrature error have been used as the Neural network features

The purpose of the last figure-Fig.5-24 is to show that even if this shallow neural network is trained on a small amount of data (5% of the total dataset), it can provide good classification accuracy. This can be that the number of neurons in the hidden layer is quite low, but still sufficient to process the input data. This means that the fewer neurons are used, the fewer input data we need.

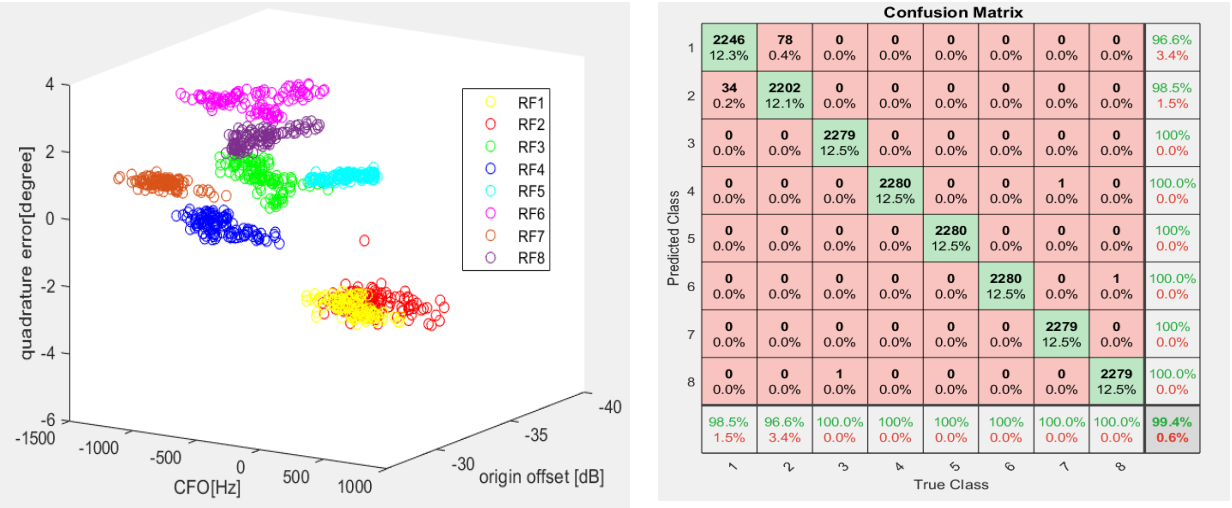


Figure.5-24: The data distribution (0.5 % of the total dataset) for the eight transmitters and their confusion matrix. The CFO, origin offset, and quadrature error have been used as the Neural network features

6. CONCLUSION

The presented work's primary goal was to classify the RF transmitters depending on their RF imperfections using a selected machine learning algorithm. In the theoretical part, we first studied the analog devices (digital to analog converter, quadrature mixer, local oscillator, and power amplifier) that are present in the RF front end. The possible impairments (such as the power amplifier nonlinearity, IQ modulator imbalance, constellation origin offset, carrier frequency offset) that can occur in the RF front-end are also studied. Then, three supervised machine learning algorithms were presented: SVM, KNN, and neural networks. A detailed explanation was about the SVM and neural network algorithms.

In the practical part, The implementation of the SVM and neural network algorithms is done in Matlab software with the use of *Statistics* and *Machine learning Toolbox*.

Three scenarios for the SVM algorithm were successively built: a binary classification using the artificial data, a binary classification using a real dataset of two individual transmitters, a multi-class classification SVM using a real dataset of three transmitters and then a multi-class classification real dataset of eight transmitters. From the obtained results according to the first scenario, it has been noticed that the SVM classifier has the best performance when the dataset is linearly separable. It has also been noticed that the performance of the SVM classifier is degraded with increased noise dispersion because the data became non-linearly separable. In the second scenario, the SVM algorithm is verified by several graphs: plots of gain imbalance versus CFO or, gain imbalance versus quadrature error, and the performance is evaluated by a ROC curve and so-called confusion matrix. In the case of using only the gain imbalance and CFO as the SVM features (non-linearly separable case), the used tuning penalty parameter *C* does not improve the overfitting. Thus, an additional imperfection (the quadrature error) has been added to the classification, making the data linearly separable. Then various ROC curves and confusion matrices are generated when classifying between RF1-RF4, RF1-RF2, and RF3-RF4 transmitter pairs. The worst SVM classifier performance was observed when classifying between RF3 and RF4 transmitters, in which case the confusion matrix indicates that the total classification accuracy is equal only to 61.9%. From the obtained results in the third scenario, the multi-class classification according to the one-vs-one algorithm is implemented. Various combinations of the transmitter features are used in two-dimensional and three-dimensional space. The selected features combination were: CFO and gain imbalance, CFO and quadrature error, CFO and origin offset, CFO and gain imbalance and quadrature error, CFO and origin offset and quadrature error. The confusion matrices have been generated for each combination of features. It has been noticed that the best dataset separation was in the case of the CFO and origin offset selected as the SVM features, while the worst case corresponds to the CFO and the gain imbalance as the SVM features (high data overlapping between RF1 and RF2, RF3 and RF4 and RF8 transmitters). The confusion matrices and their results have been interpreted by explaining how the one-vs-one multi-class algorithm works.

The neural network algorithm used in our work was the shallow neural network. At first, the optimal number of hidden layer neurons is determined, which was 20 neurons. Then, similar to SVM, various combinations of the eight transmitter features are used in two-dimensional and three-dimensional space to know how the performance of the neural

network classifier varies as the selected features change and determine which feature is the best to get robust classification between all transmitters. In the case of 2D features representation, it has been noticed that the worst neural network performance was when we have used the CFO and the gain imbalance as the neural network features. The SVM classifier misclassified especially the third and the fourth transmitter (about 60 to 70 % of the observations were misclassified). The best performance was observed when we have used the CFO and the origin offset as the neural network features. The total classification accuracy was 96.8% (all transmitters are classified correctly with a few classification errors). In the case of 3D features representation, the best performance was observed when using the CFO, the origin offset, and quadrature error as the neural network features (the total classification accuracy was 99.5%).

Thus, the origin offset feature plays the most important role in the robust classification between all transmitters for SVM and neural networks algorithms.

To conclude, the neural network classifier seems, in our case, to give more clear results than its SVM counterpart. However, the interpretation of the one-vs-one algorithm is quite complicated and its performance seems still not to be good enough for practical implementation, especially due to the selected voting strategy. There is a potential for its improvement by using advanced techniques, such as presented in [22].

Literature

[1] POSPISIL, Martin, Roman MARSALEK a Tomas GOTTHANS. Wireless device classification through transmitter imperfections — Evaluation of performance degradation due to the chip heating. *In: 2017 IEEE Radio and Wireless Symposium (RWS)* [online]. IEEE, 2017, 2017, s. 169-172 [cit. 2021-5-19]. ISBN 978-1-5090-3446-8. Dostupné z: doi:10.1109/RWS.2017.7885978

[2] GARCIA-HERNANDEZ, Martin, Alfonso PRIETO-GUERRERO a Gerardo A. LAGUNA-SANCHEZ. Survey on Compensation for Analog Front End Imperfections by Means of Adaptive Digital Front End for On-Chip OFDM Wireless Transmitters. In: *2011 IEEE Electronics, Robotics and Automotive Mechanics Conference* [online]. IEEE, 2011, s. 343-348 [cit. 2021-5-14]. ISBN 978-1-4577-1879-3. Dostupné z: doi:10.1109/CERMA.2011.63

[3] POSPISIL, Martin, Roman MARSALEK a Jitka POMENKOVA. Wireless device authentication through transmitter imperfections — measurement and classification. In: *2013 IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)* [online]. IEEE, 2013, 2013, s. 497-501 [cit. 2021-5-19]. ISBN 978-1-4673-6235-1. Dostupné z: doi:10.1109/PIMRC.2013.6666187

[4] MORELLI, Michele a Marco MORETTI. *Carrier Frequency Offset Estimation for OFDM* Direct-Conversion Receivers. *IEEE Transactions on Wireless Communications* [online]. 2012, **11**(7), 2670-2679 [cit. 2021-5-14]. ISSN 1536-1276. Dostupné z: doi:10.1109/TWC.2012.051512.120057

[5] HANNA, Samer S. a Danijela CABRIC. Deep learning Based Transmitter Identification using Power Amplifier Nonlinearity. In: *2019 International Conference on Computing, Networking and Communications* (ICNC) [online]. IEEE, 2019, 2019, s. 674-680 [cit. 2021-5-14]. ISBN 978-1-5386-9223-3. Dostupné z: doi:10.1109/ICCNC.2019.8685569

[6] BELL, Jason. *Machine Learning* [online]. Indianapolis, IN, USA: John Wiley & Sons, 2014 [cit. 2021-5-14]. ISBN 9781119183464. Dostupné z: doi:10.1002/9781119183464

[7] KECMAN, V. Support Vector Machines – An Introduction. WANG, Lipo, ed. *Support Vector Machines: Theory and Applications* [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, 2005-4-22, s. 1-47 [cit. 2021-5-14]. Studies in Fuzziness and Soft Computing. ISBN 978-3-540-24388-5. Dostupné z: doi:10.1007/10984697_1

[8] *Support Vector Machines* [online]. New York, NY: Springer New York, 2008 [cit. 2021-5-14]. Information Science and Statistics. ISBN 978-0-387-77241-7. Dostupné z: doi:10.1007/978-0-387-77242-4

[9] PATLE, A. a D. S. CHOUHAN. SVM kernel functions for classification. In: *2013 International Conference on Advances in Technology and Engineering*

(ICATE) [online]. IEEE, 2013, 2013, s. 1-9 [cit. 2021-5-15]. ISBN 978-1-4673-5619-0. Dostupné z: doi:10.1109/ICAdTE.2013.6524743

[10] McGregor, M. (2020, July 2). SVM Machine Learning Tutorial – *What is the Support Vector Machine Algorithm, Explained with Code Examples*. freeCodeCamp.org. https://www.freecodecamp.org/news/svm-machine-learning-tutorial-what-is-the-support-vector-machine-algorithm-explained-with-code-examples/.

[11] HACHIMI, Marouane, Georges KADDOUM, Ghyslain GAGNON a Poulmanogo ILLY. Multi-stage Jamming Attacks Detection using Deep Learning Combined with Kernelized Support Vector Machine in 5G Cloud Radio Access Networks. In: *2020 International Symposium on Networks, Computers and Communications (ISNCC)* [online]. IEEE, 2020, 2020-10-20, s. 1-5 [cit. 2021-5-15]. ISBN 978-1-7281-5628-6. Dostupné z: doi:10.1109/ISNCC49221.2020.9297290

[12] CHAMASEMANI, Fereshteh Falah a Yashwant Prasad SINGH. Multi-class Support Vector Machine (SVM) Classifiers -- An Application in Hypothyroid Detection and Classification. In: *2011 Sixth International Conference on Bio-Inspired Computing: Theories and Applications* [online]. IEEE, 2011, 2011, s. 351-356 [cit. 2021-5-15]. ISBN 978-1-4577-1092-6. Dostupné z: doi:10.1109/BIC-TA.2011.51

[13] Baeldung. (2020, December 9). *Multiclass Classification Using Support Vector Machines. Baeldung on Computer Science*. https://www.baeldung.com/cs/svm-multiclass-classification.

[14] ZHANG, Zhongheng. Introduction to machine learning: k-nearest *neighbors*. *Annals of Translational Medicine* [online]. 2016, **4**(11), 218-218 [cit. 2021-5-15]. ISSN 23055839. Dostupné z: doi:10.21037/atm.2016.03.37

[15] Srivastava, T. S. T. (2020, October 18). *K Nearest Neighbor: KNN Algorithm:* KNN iPython & R. Analytics Vidhya. https://www.analyticsvidhya.com /blog/-2018/03/introduction-k-neighbours-algorithm-clustering/

[16] KOUTSOUKAS, Alexios, Keith J. MONAGHAN, Xiaoli LI a Jun HUAN. Deep-learning: investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data. *Journal of Cheminformatics* [online]. 2017, 9(1) [cit. 2021-5-15]. ISSN 1758-2946. Dostupné z: doi:10.1186/s13321-017-0226-y

[17] AGGARWAL, Charu C. Machine Learning with Shallow Neural Networks. AGGARWAL, Charu C. *Neural Networks and Deep Learning* [online]. Cham: Springer International Publishing, 2018, 2018-08-26, s. 53-104 [cit. 2021-5-15]. ISBN 978-3-319-94462-3. Dostupné z: doi:10.1007/978-3-319-94463-0_2

[18] WANG, Xueli, Yufeng ZHANG, Hongxin ZHANG, Xiaofeng WEI a Guangyuan WANG. Identification and authentication for wireless transmission security based on RF-DNA fingerprint. *EURASIP Journal on Wireless Communications and Networking* [online]. 2019, 2019 (1) [cit. 2021-5-15]. ISSN 1687-1499. Dostupné z: doi:10.1186/s13638-019-1544-8

[19] POSPISIL, Martin, Roman MARSALEK a Tomas GOTTHANS. Wireless device classification through transmitter imperfections — Evaluation of performance degradation due to the chip heating. In: *2017 IEEE Radio and Wireless Symposium (RWS)* [online]. IEEE, 2017, 2017, s. 169-172 [cit. 2021-5-15]. ISBN 978-1-5090-3446-8. Dostupné z: doi:10.1109/RWS.2017.7885978

[20] LORENA, Ana Carolina, André C. P. L. F. DE CARVALHO a João M. P. GAMA. A review on the combination of binary classifiers in multiclass problems. *Artificial Intelligence Review* [online]. 2008, **30**(1-4), 19-37 [cit. 2021-5-15]. ISSN 0269-2821. Dostupné z: doi:10.1007/s10462-009-9114-9

[21] an Demirkesen, Hocine Cherifi. An Evaluation of Divide-and-Combine Strategies for Image Categorization by Multi-Class Support Vector Machines. *23rd International Symposium on Computer and Information Sciences*, 2008. ISCIS '08, Oct 2008, Istanbul, Turkey. pp.1 - 6, ff10.1109/ISCIS.2008.4717904ff. ffhal-00612239f

[22] Ting-Fan Wu, Chih-Jen Lin, Ruby C. Weng. (2004, April 8). *Probability Estimates for Multi-class Classification by Pairwise Coupling*. https://www.csie.ntu.edu.tw/~cjlin/papers/svmprob/svmprob.pdf.

[23] KUMAR, M. Arun a M. GOPAL. A comparison study on multiple binary-class SVM methods for unilabel text categorization. *Pattern Recognition Letters* [online]. 2010, 31(11), 1437-1444 [cit. 2021-5-15]. ISSN 01678655. Dostupné z: doi:10.1016/j.patrec.2010.02.015

List of appendixes

Appendix 1 - Brief guide for running the generated codes 55

Appendix1- Brief guide for running the generated codes

The aim of the codes provided in the zip file (The file attached to the main thesis file) is to create SVM and neural network algorithms for classifying the RF transmitters.

Once opened the zip file, you will see 6 folders:

The first folder, which is ***Binary SVM classification with the artificial data*.** Just run it and you will see the results.

The second folder, which is the ***Binary SVM classification with real data*** is divided into these subcodes :

- BinarySVMClassification.m: Run this subcode first. This subcode applies the binary SVM classification and generates the ROC curve to evaluate the classifier.
- DataSepar.m: this subcode divides the dataset into training data and testing data.

The third folder: which is ***Multiclass SVM classification with real data*** is divided into various subcodes :

- MultiSVM8Transmitters.m: start running this subcode first.
- Confusiontool.m: the outputs of the Binary classifiers are used to plot the confusion matrix. The confusion matrix is plotted with the use of a toolbox in Matlab
- SVMTrain.m: This subcode creates the SVM multi-class classifier.
- Draw.m: This subcode plots the data of the selected transmitters.
- DataSepar.m: This subcode divides the dataset into testing data and training data.

The fourth folder, which is the ***Multiclass SVM classification with real data according to MaxWins*** is divided into various subcodes :

- MultiClass_Segments.m: start running this subcode first. The subcode applies the multi-class SVM classification. The confusion matrix is done according to the MaxWins strategy.
- SVMTestFinel.m: This subcode applies the predict to the data and calculates the probability of occurrence classes.
- SVMTrain.m
- Draw.m
- DataSepar.m

The fifth folder, which is ***Multiclass SVM classification with the optimal hyperplanes***, is similar to the third folder. The difference between them is in this folder; we also plot the optimal hyperplanes that separate the classes

The sixth folder, which is ***Neural network classification*** with the real data is divided into these subcodes

- NeuralNetworkClassification.m: start running this subcode first. The subcode applies the classification according to the neural network algorithm, It determines the optimal number of neurons in the hidden layer. It also plots the confusion matrix.
- DataSepar.m
- uspesnost.m: calculate the success of the Neural network classifier according to the number of neurons in the hidden layer.

The last folder, which is ***The measured data of the transmitters***, contains the measured data of the eighth transmitters that we want to use for classifying. You can select the file path on your computer to get to this folder.